CONTENTS

SL NO.	TOPIC	PAGE NO.
1.	INTRODUCTION	8-10
2.	SCOPE OF THE PROJECT	11-12
3.	PROBLEM ANALYSIS AND COST ANALYSIS	13-16
4.	THEORATICAL BACKGROUND	17
5.	SOFTWARE REQUIREMENTS SPECIFICATIONS Introduction General Description External Interface Requirements Design Constraints	18-25
6.	DESIGN ERD DFD	26-27
7.	DATABASE AND TABLE DETAILS	28
8.	INTERFACE DESIGN AND CODE	29-56
9.	SYSTEM TESTING	57-58
10.	FUTURE SCOPE OF THE PROJECT	59-60
11.	CONCLUSION	61
12.	BIBLIOGRAPHY	62

1. INTRODUCTION:

Project Overview:

The landscape of wedding planning is evolving rapidly, driven by technological innovations and the changing needs of modern couples. The introduction of wedding plan management websites marks a significant leap forward in how couples approach the intricate process of organizing their special day. These websites serve as comprehensive platforms that streamline and simplify every aspect of wedding planning, offering tools for budgeting, guest list management, RSVP tracking, vendor coordination, and much more.

Project Description:

A wedding plan management website is a comprehensive online platform designed to streamline and enhance the wedding planning process. These websites are equipped with a multitude of features and tools that cater to every aspect of organizing a wedding, making the experience more efficient, personalized, and enjoyable for couples.

Key Features:

1. Comprehensive Planning Tools:

Budget Management: Tools to create, monitor, and adjust wedding budgets, ensuring couples stay on track financially. Guest List Management: Easy-to-use features for compiling guest lists, sending invitations, tracking RSVPs, and managing seating arrangements. Timeline Creation: Interactive timelines to schedule and track all wedding-related tasks and milestones.

2. Vendor Coordination:

Vendor Directories: Extensive listings of vendors with reviews, portfolios, pricing information, and availability calendars.Booking and Payment: Integrated systems for booking vendors and making secure payments directly through the website.

3. Advanced Visualization:

Virtual and Augmented Reality: VR and AR tools for virtual venue tours and visualizing wedding setups, allowing couples to make informed decisions from anywhere. Design Tools: Customizable templates for invitations, thank-you notes, and other wedding stationery.

4. Personalized Recommendations:

AI-Driven Suggestions: Artificial intelligence to provide tailored recommendations for venues, vendors, themes, and other elements based on user preferences and behavior.

5. Sustainability Features:

Eco-Friendly Options: Resources and tools to help plan sustainable weddings, including green venues, recyclable materials, and minimal waste strategies. Carbon Footprint Calculators: Tools to measure and offset the environmental impact of the wedding.

6. Inclusivity and Accessibility:

Diverse Resources: Support and resources for LGBTQ+ weddings, multicultural ceremonies, and various religious traditions. Accessibility Features: Ensuring the platform is usable by individuals with disabilities.

7. Community and Support:

Forums and Groups: Online communities for sharing experiences, seeking advice, and connecting with other couples. Expert

Consultations: Access to wedding planning experts through articles, webinars, and one-on-one consultations.

8. Mobile Optimization:

Responsive Design: Websites optimized for mobile devices, ensuring users can plan their wedding anytime, anywhere.

Dedicated Apps: Mobile applications offering the full range of planning tools and features for on-the-go convenience.

Benefits:

Efficiency: Streamlines the planning process by consolidating all tools and resources in one place.

Personalization: Offers customized solutions and recommendations that match the unique preferences of each couple.

Accessibility: Makes wedding planning accessible to a broader audience, including those with specific cultural or accessibility needs.

Support: Provides a support system through expert advice and community interactions, reducing the stress associated with planning a wedding.

In summary, wedding plan management websites are transforming how couples plan their weddings, offering a seamless, personalized, and efficient experience that adapts to modern needs and values.

2. SCOPE OF THE PROJECT

The scope of a wedding plan management project encompasses the comprehensive development and implementation of an online platform designed to streamline and enhance the wedding planning process. This scope includes several key areas:

1. Project Objectives :
Develop a user-friendly website with comprehensive wedding planning tools. Integrate advanced technologies such as AI, VR, and AR to enhance user experience. Ensure the platform supports sustainability and inclusivity. Create a seamless, mobile-optimized experience. Provide robust vendor coordination and marketplace features.

2. Core Features :
Budget Management: Tools for creating and tracking wedding budgets, with expense monitoring and financial reporting.
Guest List Management: Features for managing guest lists, sending invitations, tracking RSVPs, and arranging seating.
Timeline Creation: Interactive timelines for scheduling and tracking all wedding-related tasks.
Vendor Directories: Comprehensive listings with reviews, portfolios, pricing, and availability of various vendors.
Booking and Payment Integration: Systems for booking vendors and making secure payments through the website.
Virtual and Augmented Reality: VR and AR tools for virtual venue tours and visualizing wedding setups.
Design Tools: Customizable templates for wedding stationery and decor.
AI-Driven Recommendations: Personalized suggestions for venues, vendors, themes, and more.

3. Additional Features :

Sustainability Tools: Resources for planning eco-friendly weddings, including green vendors and carbon footprint calculators.

Inclusivity Resources: Support for LGBTQ+ weddings, multicultural ceremonies, and various religious traditions.

Community and Support: Forums, groups, and access to expert consultations for advice and support.

4. User Experience and Interface:

Intuitive Design: A user-friendly interface that simplifies navigation and enhances engagement.

Personalization: Customizable options to tailor the planning experience to individual user preferences.

Accessibility: Features that ensure the platform is usable by individuals with disabilities.

5. Technology Stack

Frontend Development: Using technologies like HTML, CSS, JavaScript, and frameworks such as React or Angular for a responsive and interactive user interface.

Backend Development: Utilizing robust backend technologies such as springboot and java on Rails to handle server-side operations, database management, and integrations.

Database Management: Implementing databases like MySQL for efficient data storage and retrieval.

Security Measures: Ensuring data privacy and security with encryption, secure payment gateways, and compliance with relevant regulations.

3. PROBLEM DEFINATION

1. Overview

Planning a wedding involves numerous tasks, including budgeting, vendor selection, guest list management, and scheduling, which can be overwhelming and time-consuming for couples. Traditional methods, such as manual spreadsheets, phone calls, and physical meetings, are often inefficient and stressful. The lack of a centralized, user-friendly platform complicates the process, leading to increased anxiety and potential oversights.

2. Key Problems

a. Complexity and Time-Consuming Processes Manual Coordination: Couples often need to coordinate with multiple vendors, manage guest lists, and track budgets manually, leading to errors and inefficiencies.
Lack of Centralization: Information is scattered across various mediums (emails, spreadsheets, paper notes), making it difficult to manage and track progress.
b. Limited Personalization and Recommendations Generic Solutions: Many existing tools offer generic templates and recommendations that do not cater to the specific tastes and preferences of couples.
Lack of Personalization: Without personalized recommendations, couples struggle to find suitable vendors, venues, and themes that match their vision.
c. Vendor Management Challenges
Inconsistent Information: Finding reliable vendors and managing contracts, payments, and communications can be inconsistent and challenging.
Limited Reviews and Portfolios: Couples often lack access to comprehensive reviews and portfolios, making it hard to assess vendor quality and fit.
d. Inadequate Visualization Tools Difficult Visualization: Visualizing the layout, decor, and overall aesthetics of the wedding is challenging without proper tools, leading to uncertainty and dissatisfaction.

Lack of Immersive Tools: The absence of VR and AR tools limits couples' ability to virtually tour venues and visualize setups.

e. Sustainability and Inclusivity Gaps Eco-Unfriendly Options: Couples seeking eco-friendly weddings face difficulties in finding and coordinating with sustainable vendors and venues.

Lack of Inclusive Resources: Existing platforms may not provide adequate resources and support for LGBTQ+ weddings, multicultural ceremonies, and various religious traditions.

f. Accessibility and Usability Issues Mobile Accessibility: Many planning tools are not optimized for mobile devices, limiting on-the-go planning capabilities.

User-Friendly Design: Platforms often lack intuitive, user-friendly designs, making the planning process more cumbersome.

3. Objectives

a. Centralized Planning Platform

Comprehensive Integration: Develop a centralized platform that consolidates all wedding planning tasks, tools, and information in one place.

Seamless Coordination: Enable seamless coordination with vendors, guest list management, and budgeting.

b. Personalized Experience

AI-Driven Recommendations: Implement AI to provide personalized recommendations for venues, vendors, and themes based on user preferences.

Customizable Templates: Offer highly customizable templates and planning tools to cater to individual tastes.

c. Enhanced Vendor Management

Robust Vendor Directory: Create an extensive directory of vendors with detailed reviews, portfolios, and pricing information.

Integrated Booking and Payments: Facilitate secure booking and payment processes directly through the platform.

d. Advanced Visualization Tools

Virtual and Augmented Reality: Incorporate VR and AR tools for virtual venue tours and visualization of wedding setups.

Design Tools: Provide interactive design tools for layout, decor, and aesthetic planning.

FEATURES OF PROPOSED WEBSITE

1. User Authentication and Profiles

- Secure user registration and login functionality.
- User profile creation and management capabilities.

2. Responsive Design

- Mobile-friendly and responsive design for seamless browsing on various devices.

3. Administrative Control Panel

- Admin panel for overseeing user management, content moderation, and analytics.

4. Security Measures

- Robust security protocols to safeguard user data and ensure secure transactions.

5. Manual work has to be reduced and Result to be received quickly
6. Increase security, speed, storing and accuracy.
7. Managing and maintaining data becomes easier and cost effective.
8. The system has been developed under as a Front-End tool and Springboot as a Back-End tool.

COST ANALYSIS

Cost Analysis using COCOMO Model :

We consider following cost drivers :

COST DRIVERS	
Required software reliability	
Complexity of the product	
Required turnabout time	
Analyst capability	
Applications experience	
Programming language experience	
Use of software tools	

Total LOC = 2000
KLOC = 2
Development Effort = 2.4 * (KLOC) 1.05 PM
= 2.4 * (2) 1.05 PM
= 4.8 * 1.05 PM
= 5.04 PM
Development Time (T_{dev}) = 2.5 * (Effort) 0.38 Months
= (12.6) 0.38 Months
= 5 Months approx.

Note : We consider our project as organic. And we follow the intermediate COCOMO estimation method to calculate effort and development time.

4. THEORETICAL BACKGROUND

In the login interface the admin or any other user will provide the Email and password to login.

In the Register interface, in case a new user wants to login then they will go through the Register button and provide Email and password and the following interface will be opened.

The following interface is the wedding Plan Management. Here user will be getting the options to select the packages listed on our website.

On selecting the Buy Now button , the user is taken to another frame where the user have to fill up the address details.

The next frame takes to the payment method where the user is asked to pay the money using card. User fills in the details and can purchase the book of his/her choice.

5. SOFTWARE REQUIREMENT SPECIFICATION

INTRODUCTION :

Wedding plan management involves the systematic organization and coordination of all aspects of a wedding to ensure a seamless and memorable event. This includes managing budgets, guest lists, venues, vendors, schedules, and logistics.

Key Components:

Budget Management: Tracking and managing expenses to stay within budget.
Guest List Management: Organizing guest information and RSVPs.
Venue Selection: Choosing and coordinating with the wedding venue.
Vendor Coordination: Managing contracts and communication with service providers.
Schedule Planning: Creating detailed timelines for the wedding day.
Decoration and Theme: Designing and implementing the wedding theme.
Logistics Management: Handling transportation and accommodations.
Contingency Planning: Preparing backup plans for unexpected issues.
Benefits:

Reduces stress for the couple.
Ensures efficient use of time and resources.
Helps maintain budget control.
Guarantees quality services.
Creates a smooth and enjoyable wedding experience.

PURPOSE :

The main purpose of the project is to keep the user interface very simple with easy login and signup protocol.
This helps users to purchase pack with an ease.
No bloatware where the user gets distracted and get a clean interface with easy to follow procedures.

SCOPE :

It may helps with buy wedding packs easily. In a very short time the collection will be very obvious, simple and sensible. The scope of wedding plan management is extensive, covering all aspects necessary to ensure a successful and memorable wedding. By addressing each component meticulously, wedding planners can effectively manage the complexities of the wedding planning process, ensuring a smooth and enjoyable experience for the couple and their guests.

REFERENCE :

- https://www.w3schools.com/java/java_intro.asp
- https://en.wikipedia.org/wiki/COCOMO
- https://www.tutorialspoint.com/java/

DEVELOPER'S RESPONSIBILITY OVERVIEW

The main responsibility of the developer is to maintain user information. There are two different modes for using this software: -

Admin Mode :

- Make sure that admin can view, assigns pack in the site, allocate , deallocate services and edit the details of the packs.
- Admin can delete record of the customer.
- The admin can edit user information records into a database.

User mode :

- Make sure that they can view his/her choice of pack and services.
- They can purchase the pack and Make payment.

GENERAL DESCRIPTION :

PRODUCT OVERVIEW :

Product function will include the following functional areas:

- Admin log into the application with username and password and phone no.
- If the administrator enters invalid name or password, then they will not be allowed to do any operations.
- In case of any user, they can also log into the application with Email , phone no. and password.
- If the user enters invalid phone number, then they will not be able to purchase packs.
- The login id must be unique.

USER CHARACTERISTICS :

- In case of a new user, they must go through the Sign Up button to create their id. They need to give their information like: - name, phone number and Email.
- In the pack purchase mode the user is needed to give the correct phone number to buy packs.
- After selecting the pack and services, User need to pay by card orselect COD mode.

General Constraints and Assumptions :

- We need to submit this project on or before 24th may, 2024. If we could get more time, then wewould have definitely improved our software by adding more facilities.

FUNCTIONAL REQUIREMENTS

General Description of Input and Output

Mainly the software is being used by the admins and the customers.

In the administrator mode - at first we need to register as an admin then we can allocate or deallocate packs in the database and assign pack and services in the respective category.

In the customer mode - at first we need to register as a customer by the giving the Email , phone no then we can

select packs and then we can step to farther procedure for the payment.

In both modes we need to register and login first to step to farther pages.

Functional Requirements :

- In the software there is a facility for the admin and customer to view the categorized pack and services.
- In the admin mode the details of the admin or the customer is also kept recorded in the database.
- They can view all the details of the allocated packs.

EXTERNAL AND INTERFACE REQUIREMENTS:

User Interface:

Interface 1: - In the registration interface the user will provide the Email, phone number and password to register.

Interface 2: - In case a customer wants to login then they will go through the sign-up button and provide a phone number, Email and the following interface will be opened.

Interface 3: - After getting registered in the database customer has to select the categorized section like “Recent PACK”, “SERVICES”, which they want to choose. They have to enter their same phone number which was given by them at the time of registration from the current logged in user.

Interface 4: - After clicking the buy now button in the previous interface a Address form will be appear then they have to fill up the address from. After that payment window will open where two payment options will be provided cash on delivery or card and the packs price will be displayed. They have to choose one of the option and pay the price.

Interface 5: - After succesfull payment they have to click the order place button to successfully place the order.

Interface 6: - User can also purchase New books by clicking New books button and Old books by clicking Old books button.

Error Messages :

1. USER REGISTRATION :

If any fields are left blank it gives and error saying **"Please fill the field".**

2. USER LOGIN :

a. If any fields are left blank it gives and error saying **"Please fill in this field and try again".**

b. If we enter the wrong Email or Password the code gives an error saying "**Password or Username is incorrect**".

DESIGN CONSTRAINTS

Hardware Requirements

- CPU 1.5GHz Dual core or higher
- RAM 4GB or higher
- Disk 10GB or higher
- Display 1280*768
- Graphics Accelerators nVidia or ATI with Support of OpenGL 1.5 or higher

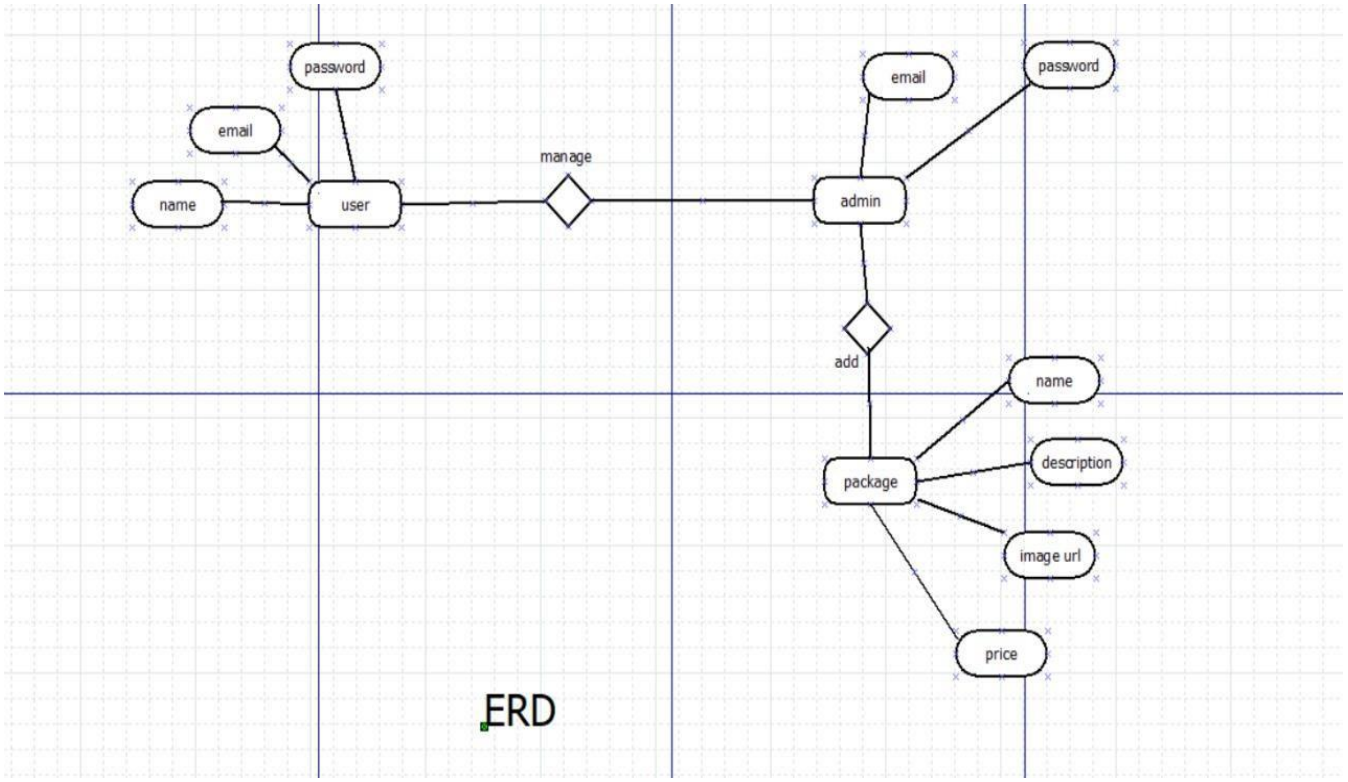
Software Requirements

- **IDE :** VS CODE, STS 4
- **FRONTEND: REACT JS**
- **BACKEND:** JAVA, SPRINGBOOT
- **OPERATING SYSTEM :** WINDOWS 7 OR HIGHER VERSION.

6.DESIGN

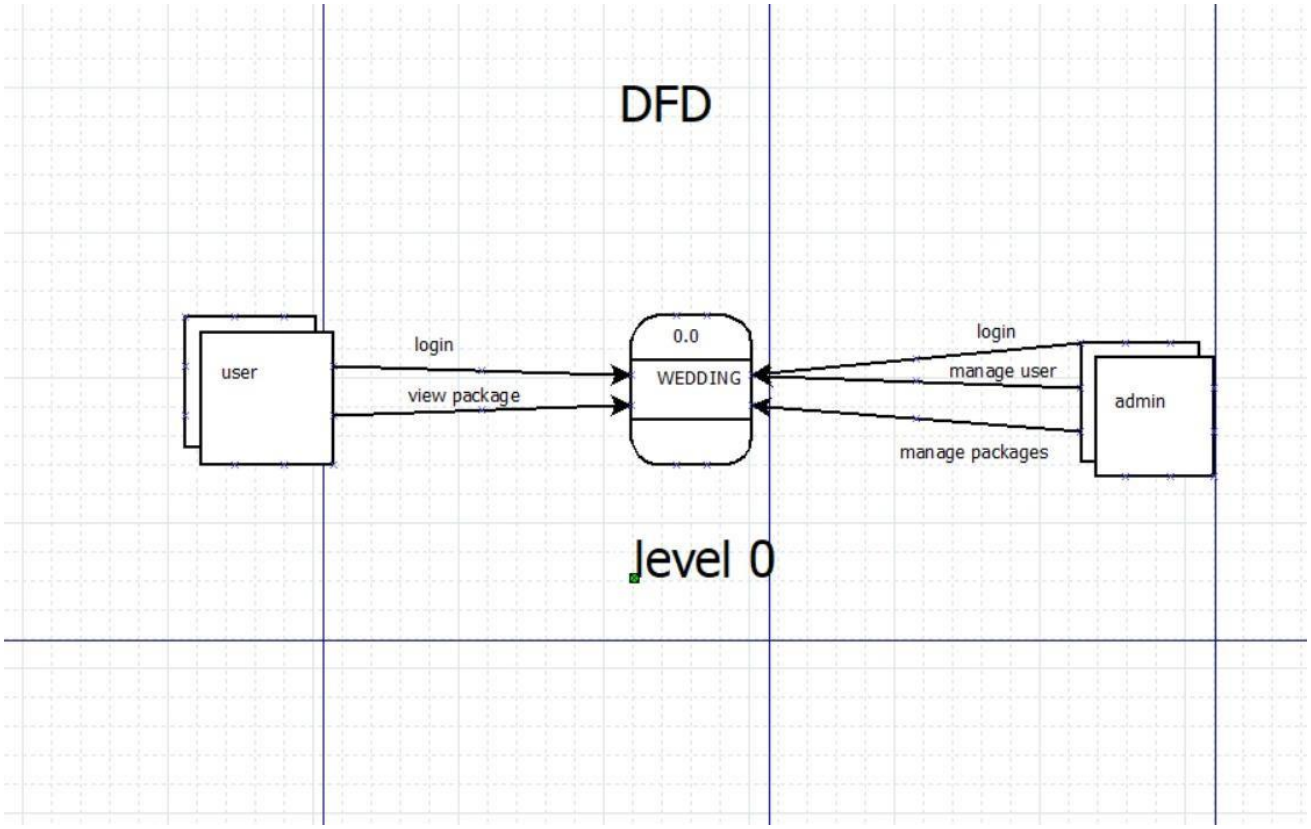
6.1 DATA DESIGN

Entity Relationship Diagram

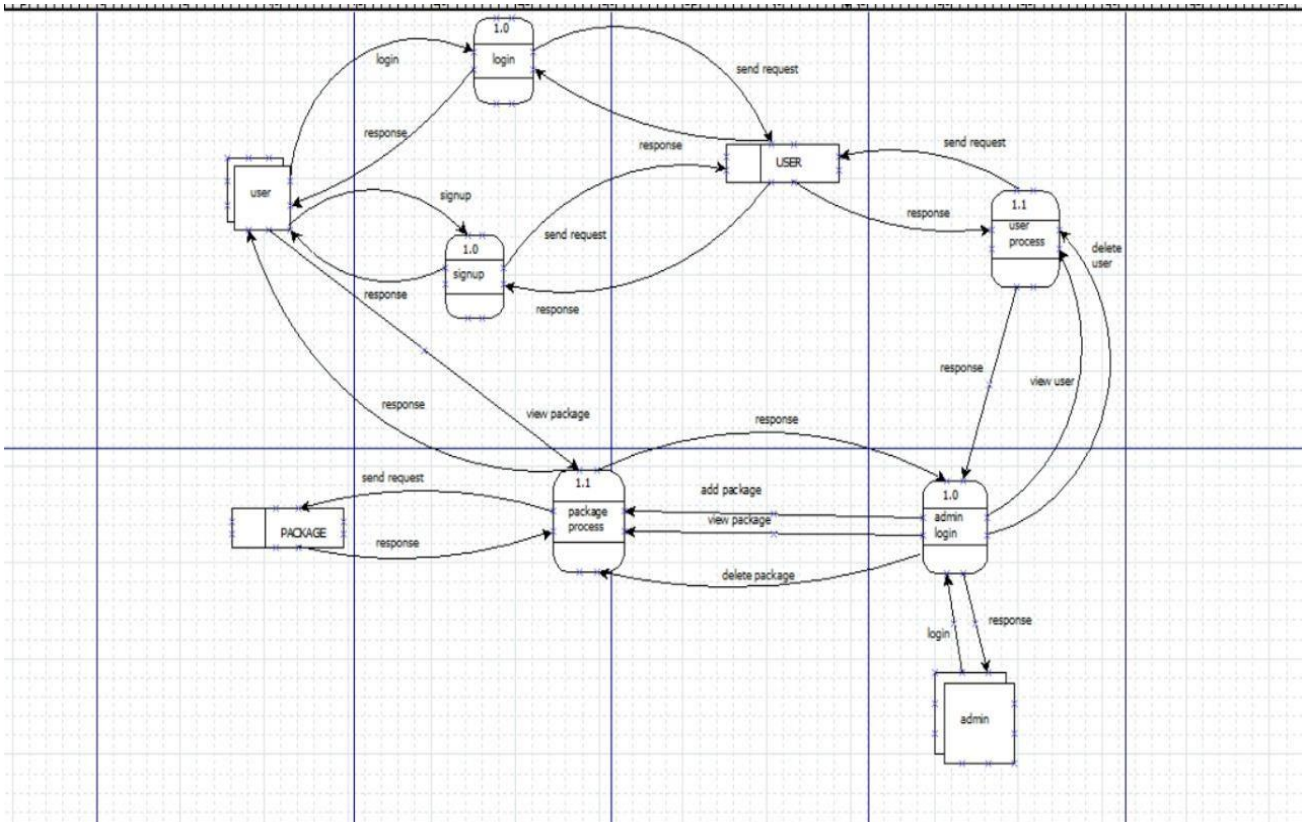


Data Flow Diagram

LEVEL 0



LEVEL 1



DATABASE AND TABLE DETAILS

Table Name**: user**

Purpose: To store details of a newly registered user.

```
mysql> SELECT * FROM user;
+----+---------------+--------------+---------------+-------------+---------------------------+----------------+----------------+
| id | product_about | product_name | product_price | product_url | email                     | name           | password       |
+----+---------------+--------------+---------------+-------------+---------------------------+----------------+----------------+
|  1 | NULL          | NULL         | NULL          | NULL        | asdfghjk@gmail.com        | Ananya Ghosh   | 1234567890     |
|  2 | NULL          | NULL         | NULL          | NULL        |                           | admin          |                |
|  3 | NULL          | NULL         | NULL          | NULL        | admin@admin.com           | admin          | admin          |
|  4 | NULL          | NULL         | NULL          | NULL        | pratyushbiswas103@gmail.com | Pratyush Biswas | 12345678     |
|  5 | NULL          | NULL         | NULL          | NULL        | palproloy92@gmail.com     | Proloy Pal     | 12345678       |
|  6 | NULL          | NULL         | NULL          | NULL        | sayantang.15@gmail.com    | Sayantan Ghosh | iloveyouananya |
|  7 | NULL          | NULL         | NULL          | NULL        | dipak69@gmail.coim        | DIPAK PANDA    | dipak69        |
+----+---------------+--------------+---------------+-------------+---------------------------+----------------+----------------+
7 rows in set (0.00 sec)
```

Table Name: admin

```
mysql> SELECT * FROM product;
+----+---------------------------------------------------------------------------------------------------------------
-----------------------------------------------------------------+-----------------------+------------------+---------------+
| id | product_about
                                                                 | product_image_url     | product_name     | product_price |
+----+---------------------------------------------------------------------------------------------------------------
-----------------------------------------------------------------+-----------------------+------------------+---------------+
| 10 | Venue: Banquet Hall (Nearby Your Location)  Catering and Menu Planning: Indulge Your Guests With A Delicious Menu Tailored To Your Tastes, Featuring
Exquisite Dishes And Impeccable Service. Decoration : Included. | https://bit.ly/3QUqpzp | Standard Package | 500000        |
+----+---------------------------------------------------------------------------------------------------------------
-----------------------------------------------------------------+-----------------------+------------------+---------------+
1 row in set (0.00 sec)
```

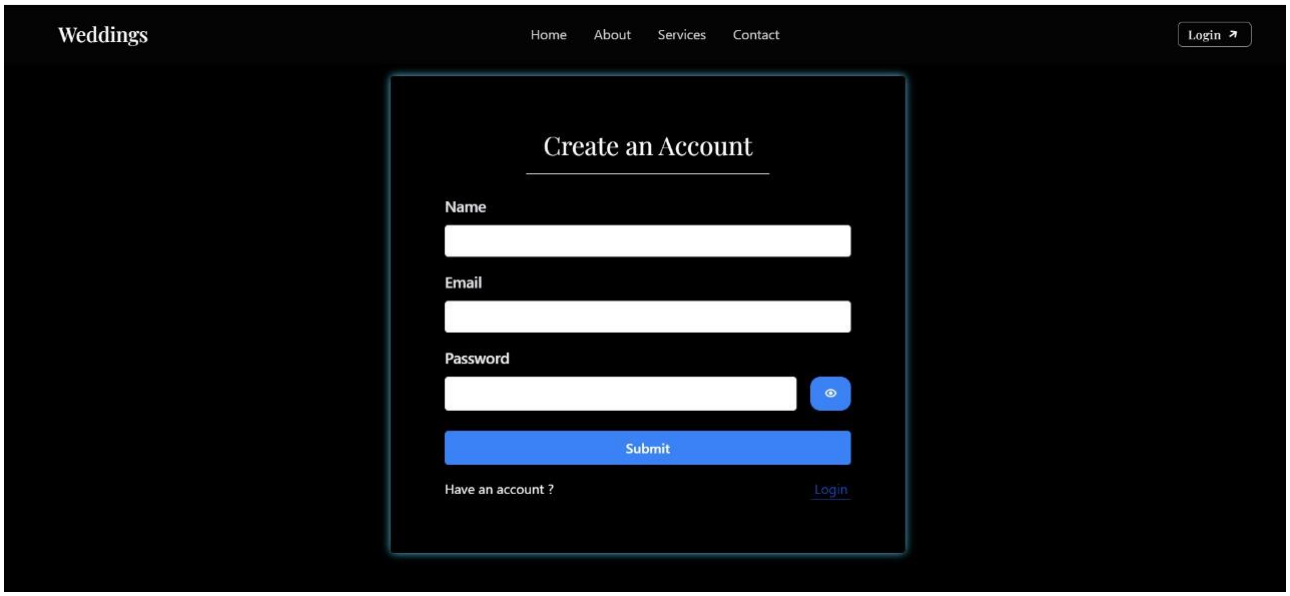
Purpose: Admin can add a package and edit the package.

Table Name: product

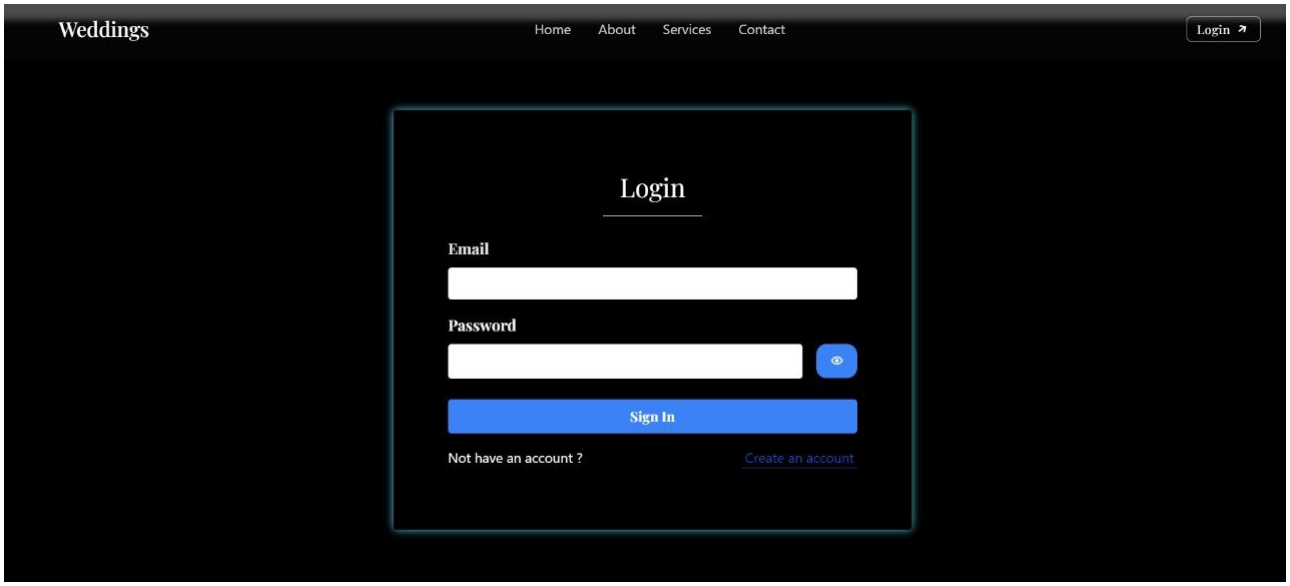
Purpose: to store details of a product details.

```
mysql> SELECT * FROM product;
+----+-------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------+--------------------------------------------------------------------+----------
-------+---------------+
| id | product_about
                                                             | product_image_url                                                  | product_na
me     | product_price |
+----+-------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------+--------------------------------------------------------------------+----------
-------+---------------+
| 10 | Venue: Banquet Hall (Nearby Your Location)  Catering and Menu Planning: Indulge Your Guests With A Delicious Menu Tailored To Your Tastes, Featuring
Exquisite Dishes And Impeccable Service. Decoration : Included. | https://bit.ly/3QUqpzp                                   | Standard P
ackage | 500000        |
| 12 | Venue: Banquet Hall (Nearby Your Location)  Catering and Menu Planning: Indulge Your Guests With A Delicious Menu Tailored To Your Tastes, Featuring
Exquisite Dishes And Impeccable Service. Decoration : Included. | https://www.images.pexels.com/photo/man-pouring-woman-yellow-flowers-1182963/ | Elite  Pac
kage   | 1000000       |
| 13 | Venue: Banquet Hall (Nearby Your Location)  Catering and Menu Planning: Indulge Your Guests With A Delicious Menu Tailored To Your Tastes, Featuring
Exquisite Dishes And Impeccable Service. Decoration : Included. | https://bit.ly/3QUqpzp                                   | Premium  P
ackage | 10000000      |
+----+-------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------+--------------------------------------------------------------------+----------
-------+---------------+
3 rows in set (0.00 sec)
```

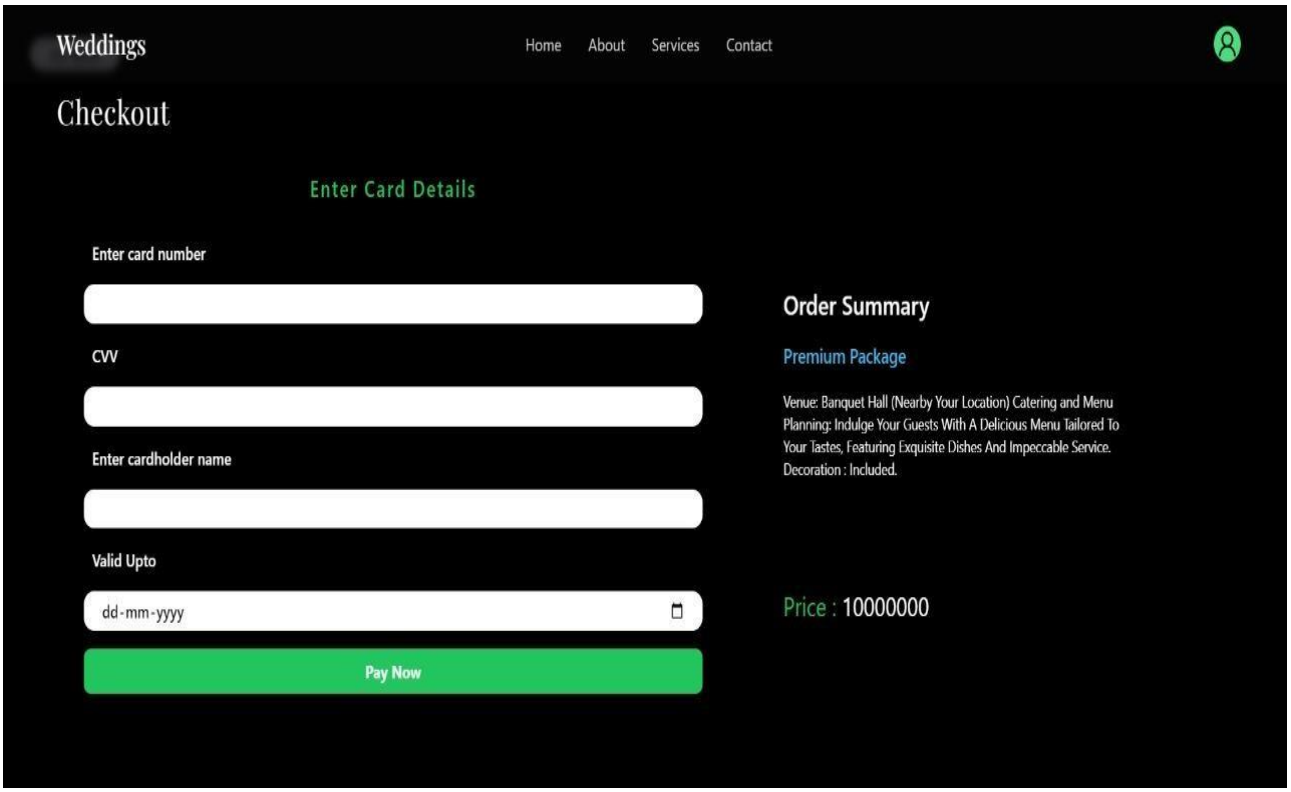
SCREEN DESIGN AND CODE




```
@PostMapping("/register")
    public ResponseEntity<?> registerUser(@RequestBody
Map<String, String> userData) {
        String email = userData.get("email");
        String name = userData.get("name");
        String password = userData.get("password");
        String productName = userData.get("ProductName");
        String productUrl = userData.get("ProductUrl");
        String productAbout = userData.get("ProductAbout");
        String productPrice = userData.get("ProductPrice");
        try {
            User user = userService.createUser(email, name,
password,productName,productUrl,productPrice,productAb
out);
            // Return user data in response
            return ResponseEntity.ok(user);
        } catch (RuntimeException e) {
            return
ResponseEntity.status(HttpStatus.BAD_REQUEST).body(e.ge
tMessage());
        }
    }
```

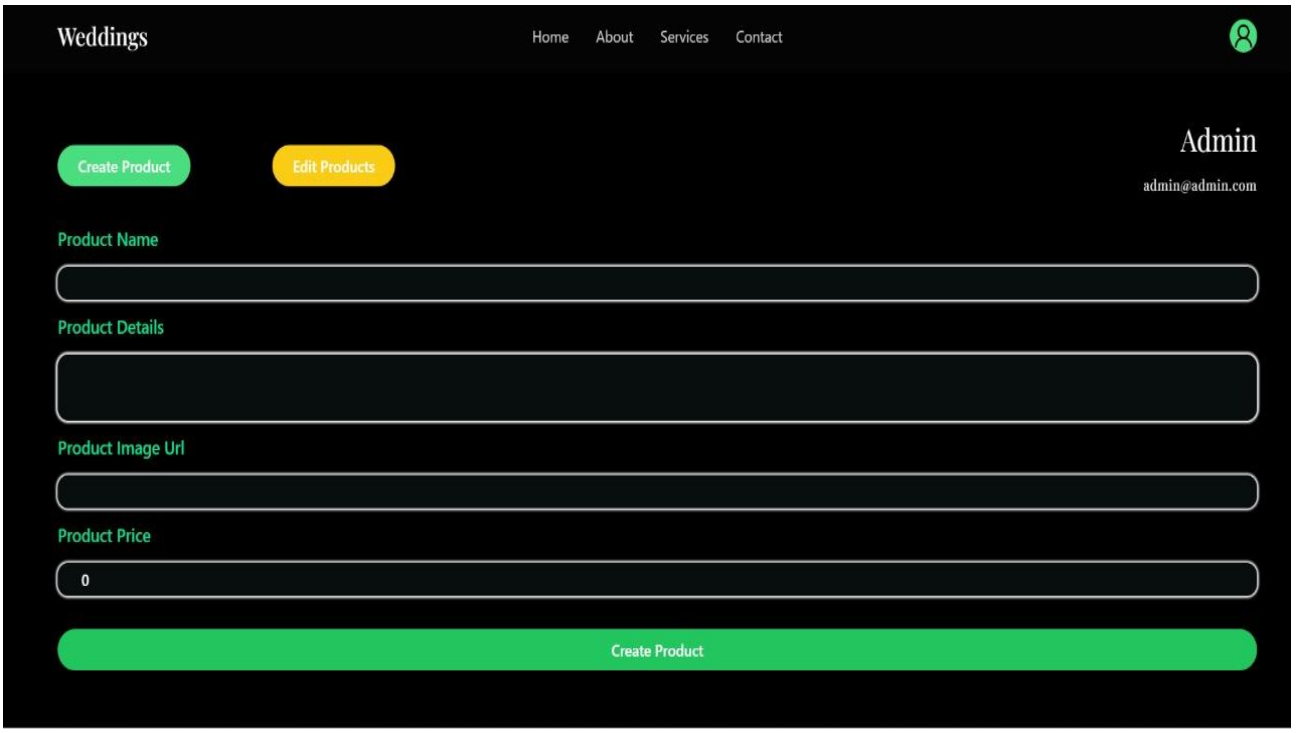



```
@PostMapping("/login")
   public ResponseEntity<?> loginUser(@RequestBody
Map<String, String> credentials) {
       String email = credentials.get("email");
       String password = credentials.get("password");

       User user = userService.getUserByEmail(email);
       if (user == null ||
!user.getPassword().equals(password)) {
           return
ResponseEntity.status(HttpStatus.UNAUTHORIZED).body("In
valid email or password");
       }

       // If login successful, return the user data
       return ResponseEntity.ok(user);
  }
```

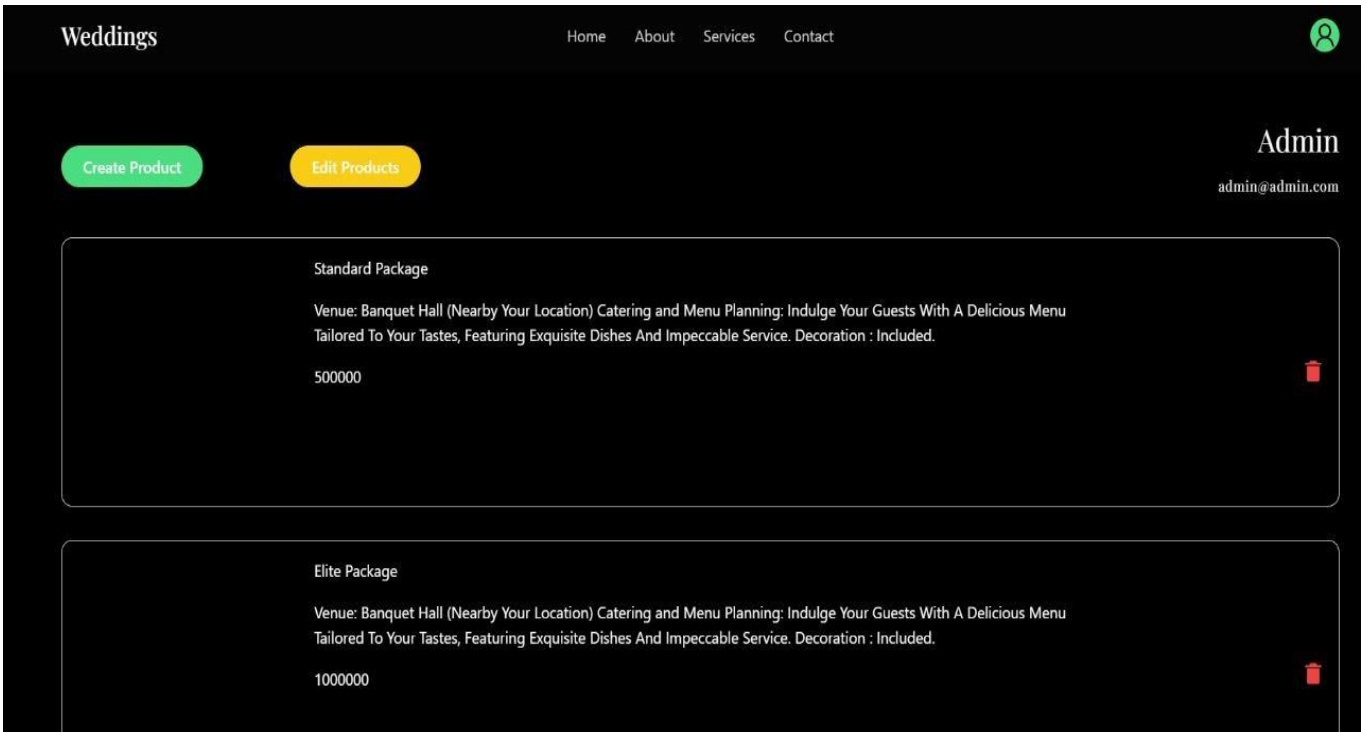



```
@PutMapping("/add-product")
    public ResponseEntity<?>
addProductToUser(@RequestBody Map< String,String>
credentials) {
        // Check if the user exists
        String email = credentials.get("email");
        String productName =
credentials.get("ProductName");
        String productAbout =
credentials.get("ProductAbout");
        String productPrice = credentials.get("ProductPrice");
        String productUrl = credentials.get("ProductUrl");
        User user = userService.getUserByEmail(email);
        if (user == null) {
            return
ResponseEntity.status(HttpStatus.NOT_FOUND).body("User
```

```
with email " + email + " not found");
        }

        try {
            // Update the user's product
            userService.updateUserProduct(email,
productName,productAbout,productPrice,productUrl);
            return ResponseEntity.ok(user);
        } catch (RuntimeException e) {
            return
ResponseEntity.status(HttpStatus.BAD_REQUEST).body(e.g
etMessage());
       }
    }


}
```

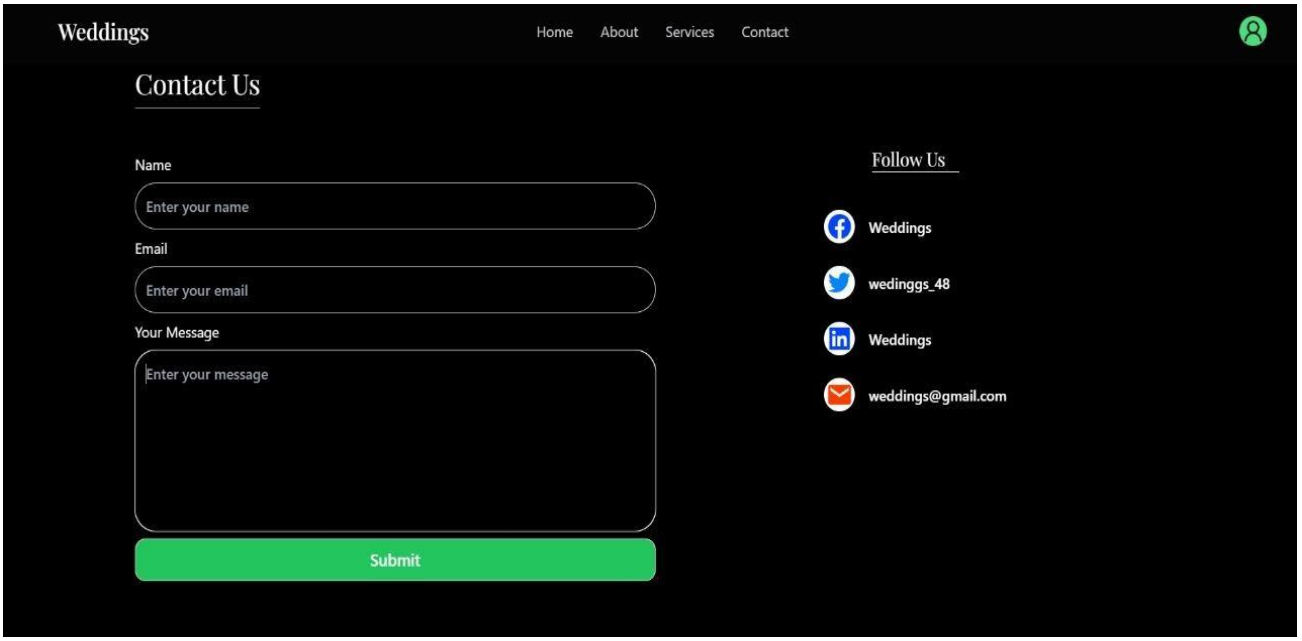



```
@RequestMapping("/api/products")
public class ProductController {

    @Autowired
    private ProductService productService;

    @PostMapping
    public ResponseEntity<Product>
addProduct(@RequestBody Product product) {
        Product newProduct =
productService.addProduct(product);
        return new ResponseEntity<>(newProduct,
HttpStatus.CREATED);
   }
```



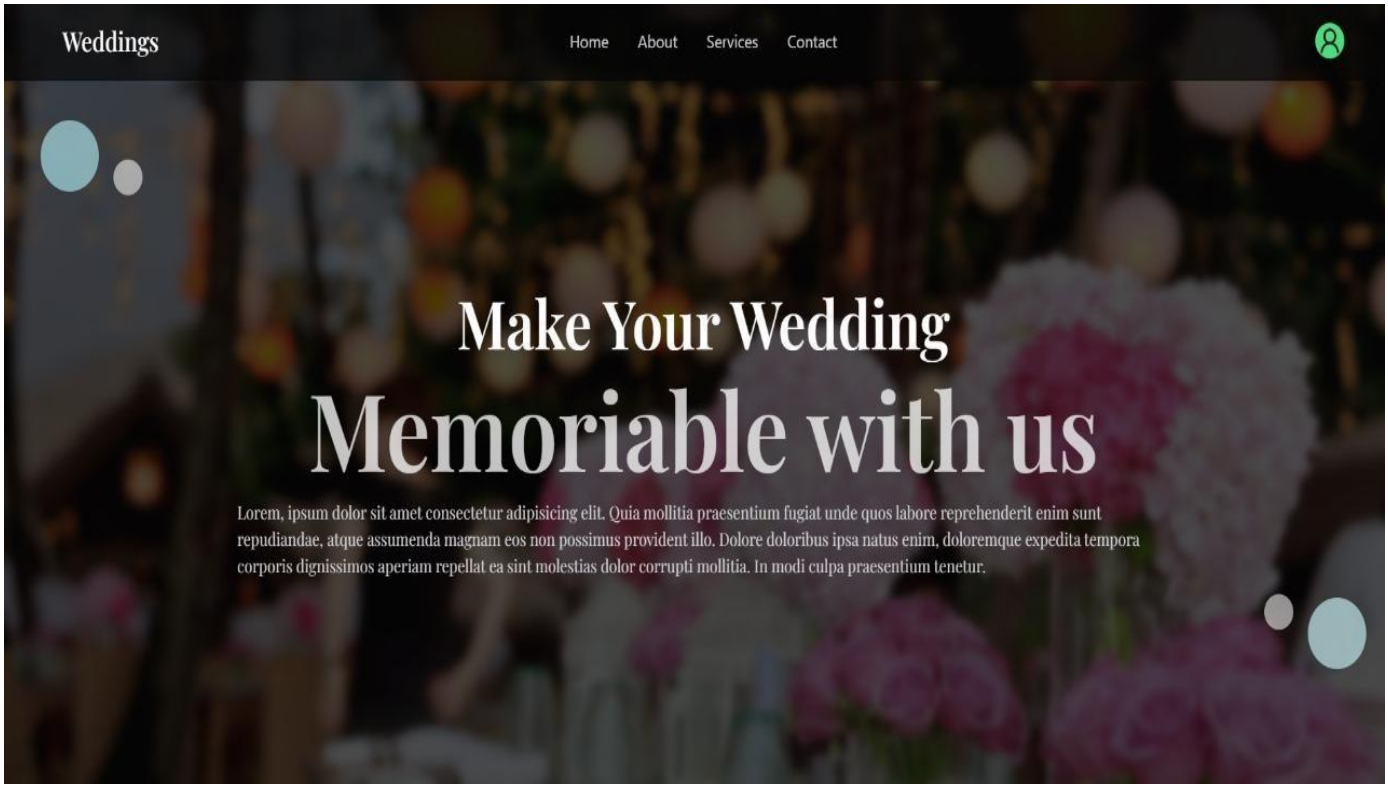

```
@RequestMapping("/api/products")
public class ProductController {

    @Autowired
    private ProductService productService;

    @PostMapping
    public ResponseEntity<Product>
addProduct(@RequestBody Product product) {
        Product newProduct =
productService.addProduct(product);
        return new ResponseEntity<>(newProduct,
HttpStatus.CREATED);
    }

    @GetMapping
    public ResponseEntity<List<Product>> getAllProducts() {
```

```
        List<Product> products =
productService.getAllProducts();
        return new ResponseEntity<>(products,
HttpStatus.OK);
    }



    @DeleteMapping("/{id}")
    public ResponseEntity<Void>
deleteProduct(@PathVariable Long id) {
        boolean deleted = productService.deleteProduct(id);
        if (deleted) {
            return new
ResponseEntity<>(HttpStatus.NO_CONTENT);
        } else {
            return new
ResponseEntity<>(HttpStatus.NOT_FOUND);
      }
   }
}
```




```
const sendEmail = (e) => {
    e.preventDefault();

    emailjs.sendForm('service_n6ypev3',
'template_7cwywm2', formRef.current, {
      publicKey: 'YawM5kwr8FiEunt1j',
    }).then(
      () => {
        toast.success('Successfully Sent!');
        // Clear form data
        setFormData({
          name: '',
          email: '',
          message: ''
        });
```

```
      },
      (error) => {
        toast.error('FAILED...', error.text);
      }
    );
};
```

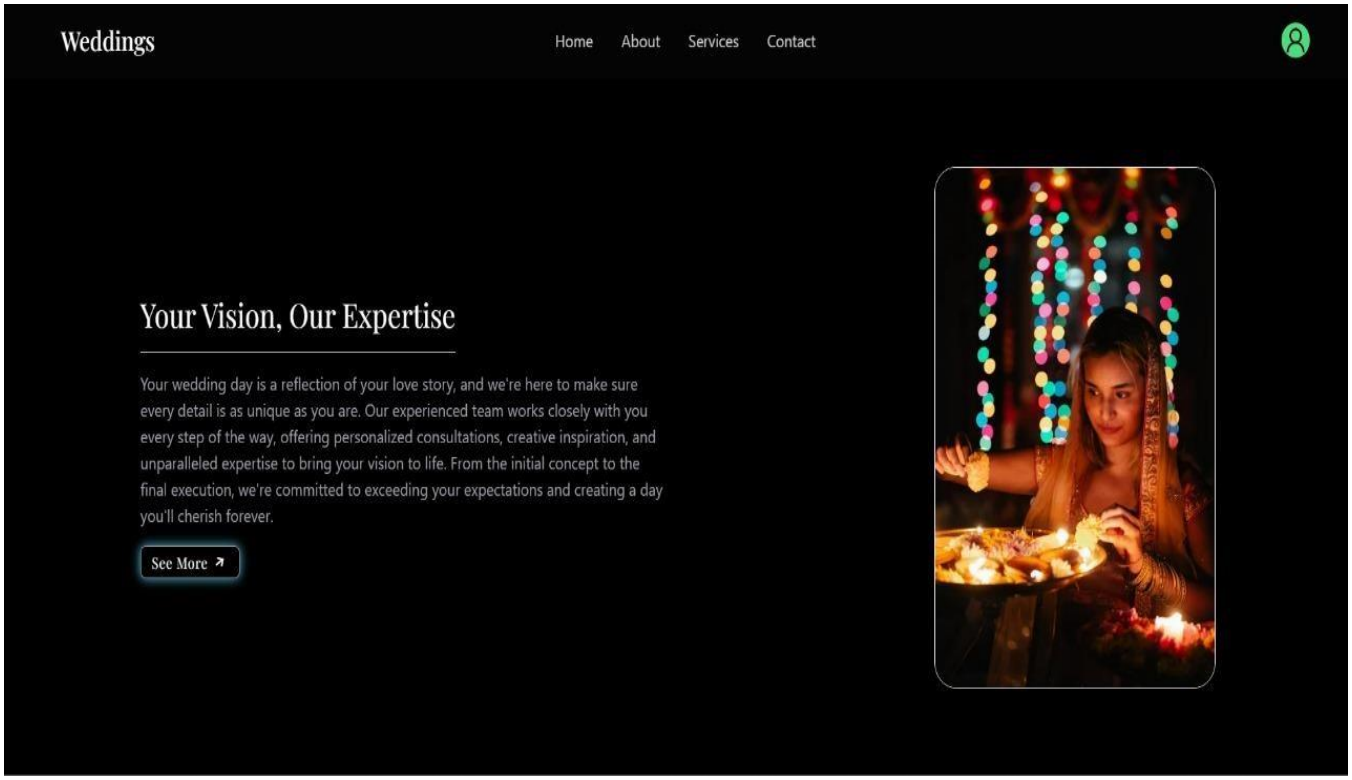



```
import React from 'react'
import { Link } from 'react-router-dom'

const Hero = ({isLogin}) => {
  return (
   <div className='  w-fit h-fit'>
 <div className=' relative w-screen h-screen bg-green-300 overflow-
hidden'>
          <img className=' absolute top-0 left-0 w-full'
src='https://images.pexels.com/photos/169190/pexels-photo-
169190.jpeg?auto=compress&cs=tinysrgb&w=1260&h=750&dpr=1'
alt="Image" />
          <div className=' w-16 h-16 absolute z-20 bg-[#ace2ebaf] top-[15vh]
```

```
left-10 rounded-full'></div>
          <div className=' w-8 h-8 absolute z-20 bg-[#ffffff8f] top-[20vh]
left-[120px] rounded-full'></div>
          <div className=' w-16 h-16 absolute z-20 bg-[#c6eef1af] bottom-
[15vh] right-10 rounded-full'></div>
          <div className=' w-8 h-8 absolute z-20 bg-[#ffffff8f] bottom-[20vh]
right-[120px] rounded-full'></div>
          <div className=' pt-[10vh] text-white flex flex-col justify-center
items-center absolute z-10 w-full h-full bgblur'>
            <h1 className=' text-6xl font-semibold headingfont text-
[#ffffff]'>Make Your Wedding</h1>
            <h1 className=' text-8xl font-semibold headingfont my-4 text-
[#f5f5f5cb]'>Memoriable with us</h1>
            <p className='headingfont w-2/3 text-zinc-300'>Lorem, ipsum
dolor sit amet consectetur adipisicing elit. Quia mollitia praesentium fugiat
unde quos labore reprehenderit enim sunt repudiandae, atque assumenda
magnam eos non possimus provident illo. Dolore doloribus ipsa natus enim,
doloremque expedita tempora corporis dignissimos aperiam repellat ea sint
molestias dolor corrupti mollitia. In modi culpa praesentium tenetur.</p>
            {isLogin?(<></>):(<><Link className=' cursor-pointer headingfont
tracking-wider text-xl hover:gap-8 duration-300 ease-linear explore border
mt-8 px-6 py-2 rounded-md gap-4 flex items-center text-white'
to="/signup">Join Today<i className='bx bx-right-arrow-alt'
></i></Link></>)}
          </div>
    </div>
    <div  className=' h-1 w-screen bg-white'></div>
    <div className=' relative w-screen h-screen bg-green-300 overflow-
hidden'>
          <img className=' absolute top-0 left-0 w-full'
src='https://images.pexels.com/photos/1128783/pexels-photo-
1128783.jpeg?auto=compress&cs=tinysrgb&w=1260&h=750&dpr=1'
alt="Image" />
          <div className=' w-16 h-16 absolute z-20 bg-[#ace2ebaf] top-[15vh]
right-10 rounded-full'></div>
          <div className=' w-8 h-8 absolute z-20 bg-[#ffffff8f] top-[20vh]
right-[120px] rounded-full'></div>
          <div className=' w-16 h-16 absolute z-20 bg-[#c6eef1af] bottom-
[15vh] left-10 rounded-full'></div>
          <div className=' w-8 h-8 absolute z-20 bg-[#ffffff8f] bottom-[20vh]
left-[120px] rounded-full'></div>
          <div className=' pt-[10vh] text-white flex flex-row pl-[15vw]
items-center absolute z-10 w-full h-full bgblur'>
            <div className=' w-fit h-[70%] rounded-xl overflow-hidden border
p-2 -translate-y-10'>
```

```
            <img className='h-full rounded-3xl'
src="https://images.pexels.com/photos/1114425/pexels-photo-
1114425.jpeg?auto=compress&cs=tinysrgb&w=600" alt="" />
            </div>
            <div className=' mx-8 flex flex-col w-2/3 items-center -
translate-y-14'>
                <h1 className=' mx-8 text-3xl pb-1 border-b headingfont
leading-[60px] px-3'>Welcome to Weddings - Where Dreams Meet
Decor</h1>
                <p className=' mx-8 headingfont tracking-wider leading-8
mt-4'>

                Congratulations! You've found the perfect partner, now let us
help you find the perfect decor for your dream wedding. At Weddings, we
specialize in turning your vision into reality, creating unforgettable
atmospheres that reflect your unique love story.
                </p>
            </div>

        </div>
    </div>
    <div  className=' h-1 w-screen bg-white'></div>
    <div className=' relative w-screen h-screen bg-green-300 overflow-
hidden'>
        <img className=' absolute top-0 left-0 w-full'
src='https://images.pexels.com/photos/12689084/pexels-photo-
12689084.jpeg?auto=compress&cs=tinysrgb&w=600' alt="Image" />
        <div className=' w-16 h-16 absolute z-20 bg-[#ace2ebaf] top-[15vh]
right-10 rounded-full'></div>
        <div className=' w-8 h-8 absolute z-20 bg-[#ffffff8f] top-[20vh]
right-[120px] rounded-full'></div>
        <div className=' w-16 h-16 absolute z-20 bg-[#c6eef1af] bottom-
[15vh] left-10 rounded-full'></div>
        <div className=' w-8 h-8 absolute z-20 bg-[#ffffff8f] bottom-[20vh]
left-[120px] rounded-full'></div>
        <div className=' pt-[10vh] text-white flex flex-row-reverse pr-
[13vw] pl-30 items-center absolute z-10 w-full h-full bgblur'>
            <div className=' w-fit h-[70%] rounded-xl overflow-hidden border
p-2 -translate-y-10'>
            <img className='h-full rounded-3xl'
src="https://images.pexels.com/photos/6479601/pexels-photo-
6479601.jpeg?auto=compress&cs=tinysrgb&w=1260&h=750&dpr=1" alt="" />
            </div>
            <div className=' mx-8 flex flex-col w-2/3 items-center -
translate-y-14'>
```

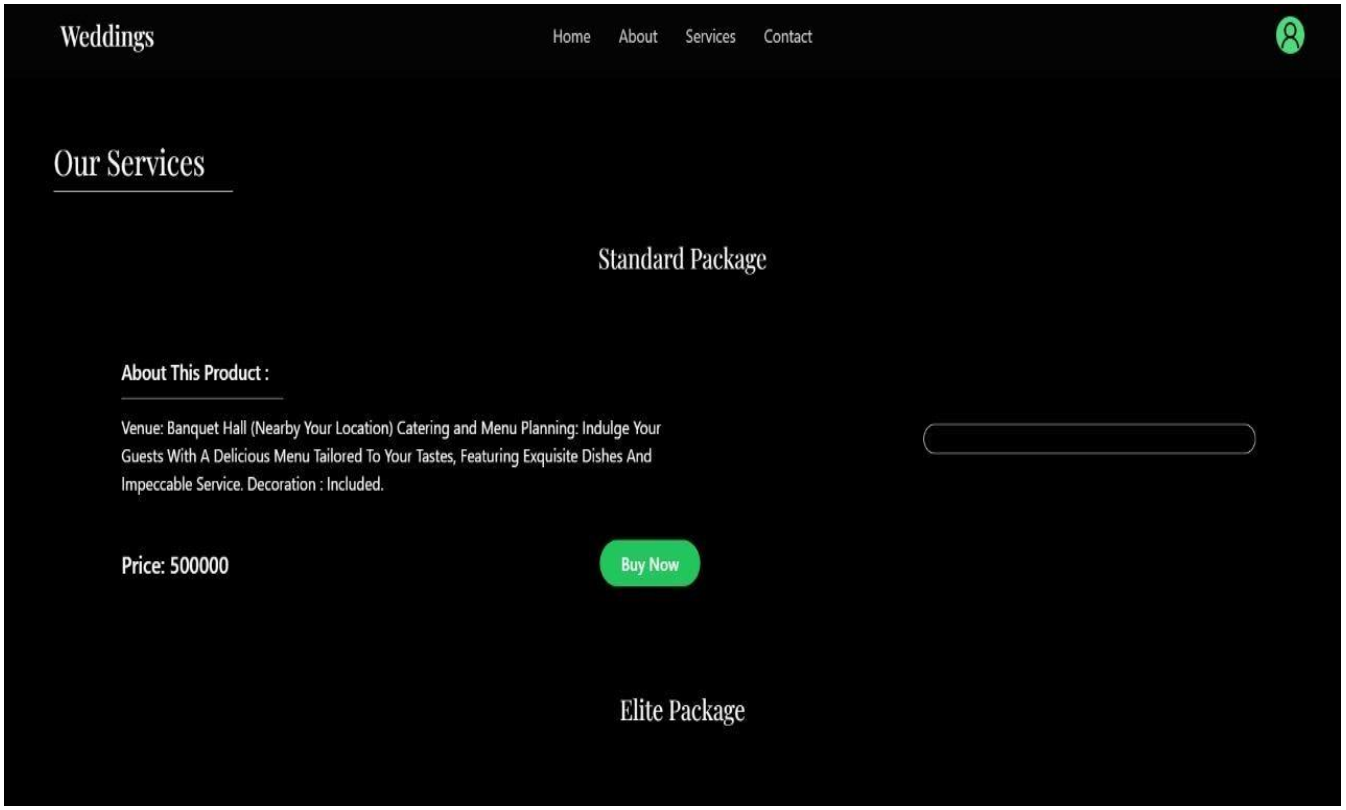
```
                <h1 className=' mx-8 pb-1 pr-6 self-start border-b text-3xl
headingfont leading-[60px]'>Elevate Your Celebration</h1>
                <p className=' mx-8 headingfont tracking-wider leading-8
mt-4'>

                Every couple is unique, and so is every wedding. Whether you
envision a classic, timeless affair or a whimsical, modern celebration, our
team is dedicated to bringing your ideas to life. From intimate gatherings
to grand soirees, we excel in crafting stunning decor that sets the stage for
your once-in-a-lifetime moment.
                </p>
            </div>

          </div>
    </div>
   </div>
  )
}

export default Hero
```




```
import React from 'react'
```

```
import Button from './Button'

const About = () => {
  return (
   <div>
     <div className='pt-[10vh] w-full h-screen bg-black flex justify-between
items-center px-[10vw] text-white'>
        <div className=' flex flex-col justify-start gap-4'>
            <h1 className=' headingfont text-3xl border-b w-fit py-4'>Your
Vision, Our Expertise</h1>
            <p className=' w-2/3 text-zinc-400'>Your wedding day is a
reflection of your love story, and we're here to make sure every detail is as
unique as you are. Our experienced team works closely with you every step
of the way, offering personalized consultations, creative inspiration, and
unparalleled expertise to bring your vision to life. From the initial concept
to the final execution, we're committed to exceeding your expectations and
creating a day you'll cherish forever.</p>
            <Button title={'See More'} shadow={true}/>
            </div>
        <img className=' h-3/4 rounded-3xl border'
src='https://images.pexels.com/photos/8887299/pexels-photo-
8887299.jpeg?auto=compress&cs=tinysrgb&w=1260&h=750&dpr=1' alt="" />
    </div>
    <div className='pt-[10vh] w-full h-screen bg-black flex flex-row-reverse
gap-4 justify-between items-center px-[10vw] text-white'>
        <div className=' flex flex-col items-start -translate-y-14 w-2/3  gap-4'>
            <h1 className=' headingfont text-3xl -translate-x-20 self-center
border-b w-fit py-4'>Unparalleled Elegance, Impeccable Details</h1>
            <p className=' w-full my-4    text-zinc-400 '>Our curated collection
of decor elements features everything from exquisite floral arrangements to
enchanting lighting designs and personalized touches that reflect your style.
With an eye for detail and a passion for perfection, we ensure that every
element of your decor harmonizes seamlessly, creating a cohesive and
breathtaking ambiance</p>
            <Button title={'Read More'} shadow={true}/>
            </div>
        <img className=' h-3/4 rounded-3xl border'
src='https://images.pexels.com/photos/712322/pexels-photo-
712322.jpeg?auto=compress&cs=tinysrgb&w=1260&h=750&dpr=1' alt="" />
    </div>
    <div className='pt-[10vh] w-full h-screen bg-black flex justify-between
items-center px-[10vw] text-white'>
        <div className=' flex flex-col justify-start gap-4'>
            <h1 className=' headingfont text-3xl border-b w-fit py-4'>Let's
Create Magic Together</h1>
            <p className=' w-2/3 text-zinc-400'>Ready to begin your journey to
```

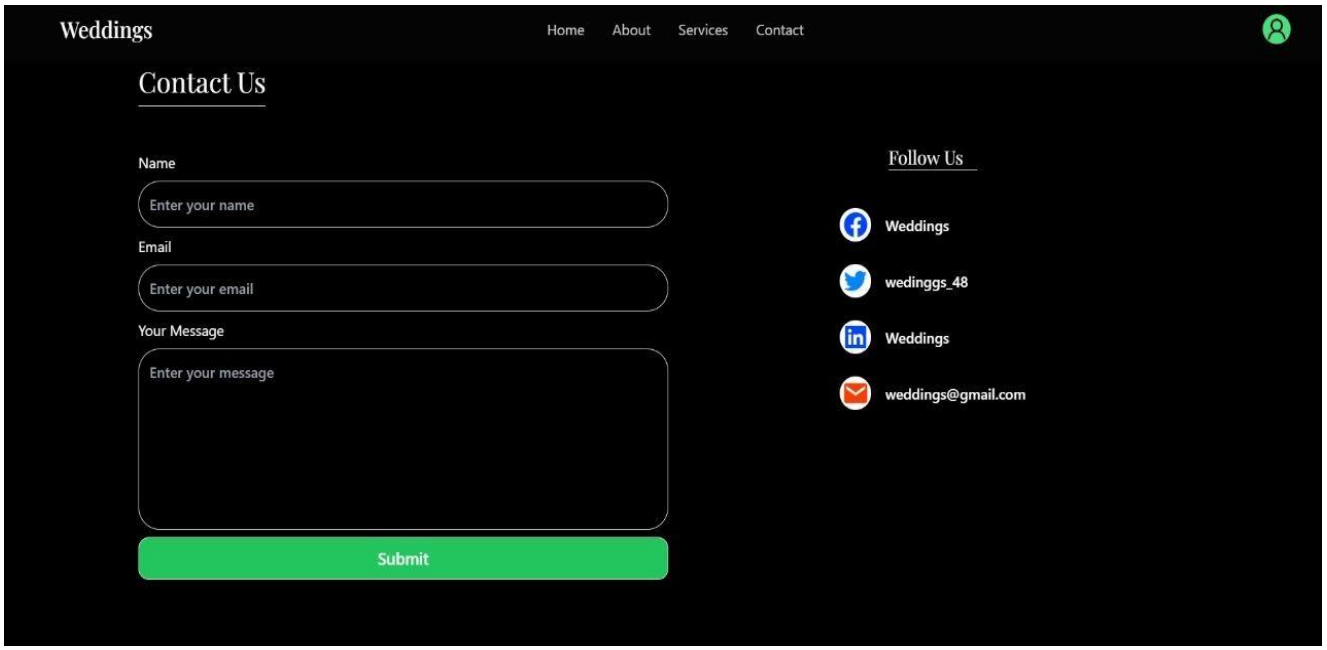
```
the perfect wedding decor? Explore our portfolio, get inspired by our gallery
of past creations, and schedule a consultation to start planning your
unforgettable celebration. At Weddings, we believe that every love story
deserves a beautiful beginning, and we can't wait to help you create
yours.</p>
            <Button title={'See More'} shadow={true}/>
            </div>
        <img className=' h-3/4 rounded-3xl border'
src='https://images.pexels.com/photos/2892274/pexels-photo-
2892274.jpeg?auto=compress&cs=tinysrgb&w=600' alt="" />
    </div>
   </div>
  )
}

export default About
```




```
import React from 'react';
import { Link } from 'react-router-dom';
import {  toast } from 'react-toastify';
import 'react-toastify/dist/ReactToastify.css';
```

```
const Services = ({ productname, loginDetails, data }) => {
  // Check if data is an array before mapping over it
  if (!Array.isArray(data)) {
    return (
      <div className='w-screen flex-col min-h-screen pt-[12vh] flex gap-10
my-10'>
        <h1 className='text-3xl headingfont ml-14 pr-8 pb-2 border-b inline w-
fit'>
          Our Services
        </h1>
        <p>No data available</p>
      </div>
    );
  }


  return (
    <div className='w-screen flex-col min-h-screen pt-[12vh] flex gap-10 my-
10'>
      <h1 className='text-3xl headingfont ml-14 pr-8 pb-2 border-b inline w-
fit'>
        Our Services
      </h1>
      {data.map((service) => (
        <div key={service.id} className='px-[5vw] mb-12'>
          <h1 className='text-2xl headingfont w-full pb-3 px-4 flex justify-
center items-center'>
            {service.productName}
          </h1>
          <div className='flex justify-between items-center w-screen h-full'>
            <div className='flex flex-col gap-3 w-1/2 mt-14 px-14'>
              <h1 className='text-lg font-medium pb-2 border-b w-fit pr-4'>
                About This Product :
              </h1>
              <p>{service.productAbout}</p>
              <div className='mt-6 flex items-center justify-between'>
                <h1 className='text-xl font-medium'>Price:
{service.productPrice}</h1>
                {loginDetails ? (
                  <Link
                    onClick={() =>  productname({
                      name: service.productName,
                      price:  service.productPrice,
                      about: service.productAbout,
                      imageurl: service.productImageUrl
```

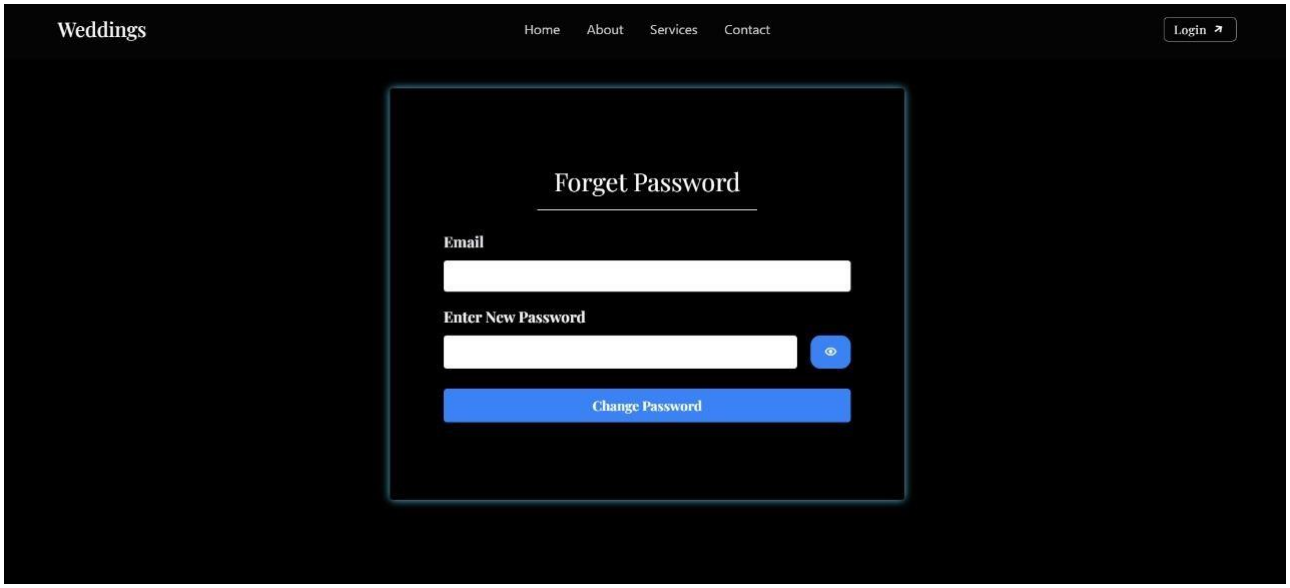
```
                })}
                to='/payment'
                className='py-2 px-6 rounded-full bg-green-500 text-white
font-medium border border-black'>
                Buy Now
              </Link>
            ) : (
              <Link
                to="/login"
                className='py-2 px-6 rounded-full bg-green-500 text-white
font-medium border border-black'>
                Buy Now
              </Link>
            )}
          </div>
        </div>
        <div className='overflow-hidden w-1/2 flex justify-center rounded-
xl p-2'>
          <img className='w-[50%] border p-3 rounded-3xl'
src={service.productImageUrl} alt='' />
        </div>
      </div>
    </div>
  ))}
 </div>
 );
}

export default Services;
```




```
import React, { useState, useRef } from 'react';
import emailjs from '@emailjs/browser';
import { ToastContainer, toast } from 'react-toastify';

const Contact = () => {
  const formRef = useRef();

  const [formData, setFormData] = useState({
    name: '',
    email: '',
    message: ''
  });

  const handleChange = (e) => {
    const { name, value } = e.target;
    setFormData(prevState => ({
      ...prevState,
      [name]: value
    }));
  };

  const sendEmail = (e) => {
    e.preventDefault();

    emailjs.sendForm('service_n6ypev3', 'template_7cwywm2',
formRef.current, {
```

```
      publicKey: 'YawM5kwr8FiEunt1j',
    }).then(
      () => {
        toast.success('Successfully Sent!');
        // Clear form data
        setFormData({
          name: '',
          email: '',
          message: ''
        });
      },
      (error) => {
        toast.error('FAILED...', error.text);
      }
    );
  };

  return (
    <div className='h-screen w-full px-[10vw] pt-2 mb-[10vh] flex flex-col
gap-10'>
      <h1 className="pt-16 pb-2 w-fit text-3xl headingfont border-b">Contact
Us</h1>
      <div className='flex w-full'>

        <form ref={formRef} className='flex flex-col w-1/2 gap-2'
onSubmit={sendEmail} method="post">
        <ToastContainer/>
          <label htmlFor="name">Name</label>
          <input className='text-white p-3 rounded-3xl bg-[#000000] border
font-medium' placeholder='Enter your name' onChange={handleChange}
value={formData.name} id='name' name="name" type="text" />
          <label htmlFor="email">Email</label>
          <input className='text-white p-3 font-medium bg-[#000000] border
rounded-3xl' placeholder='Enter your email' value={formData.email}
onChange={handleChange} id='email' name="email" type="email" />
          <label htmlFor="message">Your Message</label>
          <textarea className='text-white bg-[#000000] border p-3 resize-none
rounded-3xl overflow-y-scroll scroller font-medium' placeholder='Enter your
message' onChange={handleChange} value={formData.message}
name="message" id="message" cols="10" rows="7"></textarea>
          <button className='bg-green-500 rounded-xl px-8 py-2 border text-lg
font-medium'>
            Submit
          </button>
        </form>
        <div className='flex gap-10 flex-col items-center w-1/2 h-full'>
```

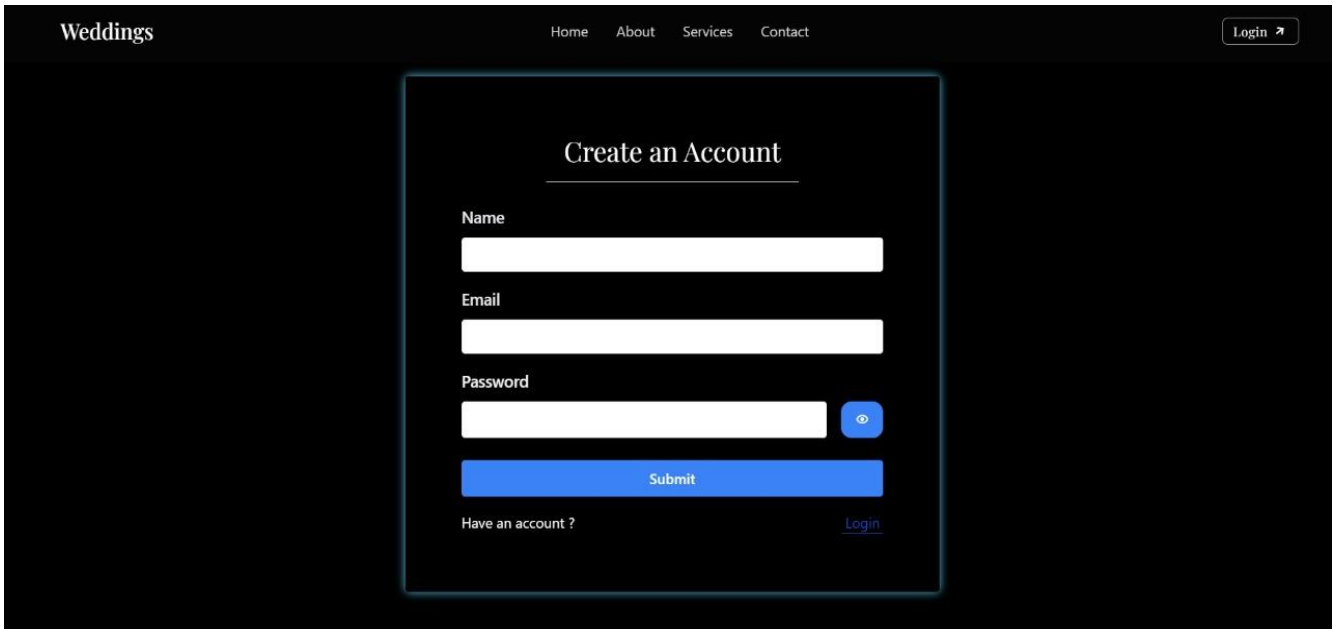
```
          <h1 className='headingfont border-b pr-4 text-xl'>
            Follow Us
          </h1>
          <div className='flex flex-col gap-6 as font-semibold'>
            <a className='flex gap-4 items-center hover:translate-x-2 duration-
200 ease-in cursor-pointer' to="#"><i className='bx bxl-facebook-circle'></i>
Weddings</a>
            <a className='flex gap-4 items-center hover:translate-x-2 duration-
200 ease-in cursor-pointer' to="#"><i className='bx bxl-twitter'></i>
wedinggs_48</a>
            <a className='flex gap-4 items-center hover:translate-x-2 duration-
200 ease-in cursor-pointer' to="#"><i className='bx bxl-linkedin-square'></i>
Weddings</a>
            <a className='flex gap-4 items-center hover:translate-x-2 duration-
200 ease-in cursor-pointer' to="#"><i className='bx bxs-envelope'></i>
weddings@gmail.com</a>
          </div>
        </div>
      </div>
    </div>
  );
};

export default Contact;
```




```
import React, { useState ,useEffect} from 'react';
import { Link, useNavigate } from 'react-router-dom'; // Import useNavigate
here
import { ToastContainer, toast } from 'react-toastify';
```

```
import 'react-toastify/dist/ReactToastify.css';

const ForgotPass = () => {
    const navigate = useNavigate();
    const [email,setEmail]=useState('')
    const[password,setPassword]= useState('')
    const [hide, setHide] = useState({
        name: 'bx bx-show',
        type: 'password'
    });


    const handlePass = () => {
        hide.name === 'bx bx-show' ? setHide({ name: 'bx bx-hide', type: 'text'
}) : setHide({ name: 'bx bx-show', type: 'password' });
    };

    const handleSubmit =(e)=>{
            e.preventDefault()
            toast.success("Password Changed Sucessfully")
            setTimeout(() => {
                navigate("/")
            }, 1500);

    }
  return (
    <div className=" loginbg flex justify-center items-center h-screen">
            <form onSubmit={handleSubmit} className=" glass bxshadow text-
white shadow-md rounded w-[40%] h-[70%] flex flex-col justify-center items-
start p-16">
                <h1 className=' self-center pb-4 mb-6 border-b px-5 headingfont
text-3xl text-white'>Forget Password</h1>
                <div className="mb-4 w-full">
                    <label className="block text-gray-200 text-lg font-bold mb-2
headingfont" htmlFor="username">
                        Email
                    </label>
                    <input
                        className="shadow appearance-none border rounded w-full
py-2 px-3 text-gray-700 leading-tight focus:outline-none focus:shadow-
outline"
                        id="username"
                        type="text"
                        placeholder=""
                        value={email.toLowerCase()}
                        onChange={(e) => setEmail(e.target.value)}
```

```
                    />
                </div>
                <div className="mb-6 w-full">
                    <label className="block text-gray-100 text-lg font-bold mb-2
headingfont" htmlFor="password">
                        Enter New Password
                    </label>
                    <div className=' flex h-fit gap-4  items-center'>
                    <input
                        className="shadow text-lg appearance-none border
rounded w-full py-2 px-3 text-gray-700 leading-tight focus:outline-none
focus:shadow-outline"
                        id="password"
                        type={hide.type}
                        placeholder=""
                        value={password}
                        onChange={(e) => setPassword(e.target.value)}
                    />
                    <button type="button" onClick={handlePass} className=' px-4
py-2 rounded-xl bg-blue-500 font-medium text-white'><i
className={hide.name}></i></button>

                    </div>
                </div>
                <div className="flex items-center justify-between w-full">
                    <button
                        className="bg-blue-500 hover:bg-blue-700 headingfont
text-white font-bold w-full py-2 px-4 rounded focus:outline-none
focus:shadow-outline"
                        type="submit"
                    >
                        Change Password
                    </button>
                    <ToastContainer/>
                </div>

            </form>
        </div>
    )
}

export default ForgotPass
```

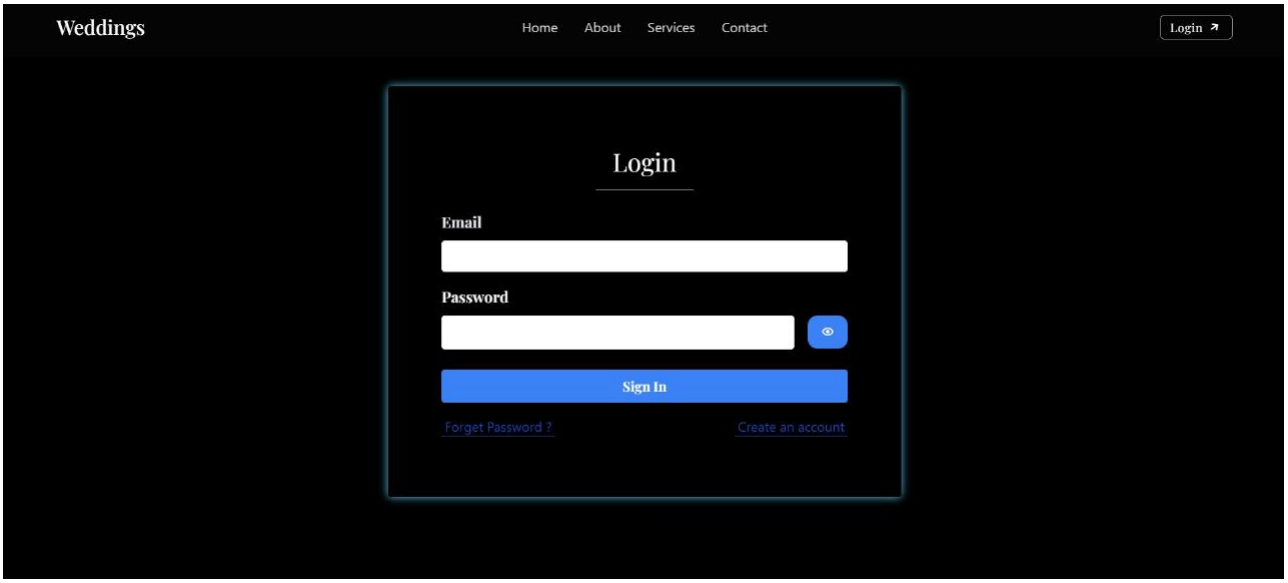



```
import React, { useState } from 'react';
import { Link ,useNavigate} from 'react-router-dom';
import { ToastContainer, toast } from 'react-toastify';
import 'react-toastify/dist/ReactToastify.css';

const Signup = ({isLogin}) => {
    const [email, setEmail] = useState('');
    const [name, setName] = useState('');
    const [password, setPassword] = useState('');
    const [hide, setHide] = useState({
        name: 'bx bx-show',
        type: 'password'
    });
    const navigate = useNavigate()

    const handleSubmit = async (e) => {
        e.preventDefault();

        try {
            const response = await fetch('http://localhost:8080/register', {
                method: 'POST',
                headers: {
                    'Content-Type': 'application/json'
                },
                body: JSON.stringify({
                    email,
                    name,
                    password
                })
```

```
            });

            if (response.ok) {
                // Registration successful, handle success
                 toast.success('Registered successfully please Log in');
                 setTimeout(() => {
                    navigate("/login")
                 }, 1000);


            } else {
                // Registration failed, handle error
                toast.error('User alredy exist please log in');
            }
        } catch (error) {
            console.error('Error registering user:', error);
            toast.error('Error registering user');
        }
    };

    const handlePass = () => {
        setHide(prevState => ({
            name: prevState.name === 'bx bx-show' ? 'bx bx-hide' : 'bx bx-show',
            type: prevState.type === 'password' ? 'text' : 'password'
        }));
    };

    return (
        <div className="loginbg flex justify-center items-center h-fit">

            <form onSubmit={handleSubmit} className="glass bxshadow text-
white shadow-md rounded w-[40%] h-[80vh] my-[12vh] flex flex-col justify-
center items-start p-16">
            <ToastContainer/>
                <h1 className='self-center pb-4 mb-6 border-b px-5 headingfont
text-3xl text-white'>Create an Account</h1>
                <div className="mb-4 w-full">
                    <label className="block text-gray-200 text-lg font-medium
mb-2" htmlFor="name">
                        Name
                    </label>
                    <input
                        className="shadow appearance-none border rounded w-full
py-2 px-3 text-gray-700 leading-tight focus:outline-none focus:shadow-
outline"
                        id="name"
```

```
                    type="text"
                    placeholder=""
                    value={name}
                    onChange={(e) => setName(e.target.value)}
                  />
                </div>
                <div className="mb-4 w-full">
                  <label className="block text-gray-200 text-lg font-medium
mb-2" htmlFor="username">
                    Email
                  </label>
                  <input
                    className="shadow appearance-none border rounded w-full
py-2 px-3 text-gray-700 leading-tight focus:outline-none focus:shadow-
outline"
                    id="username"
                    type="text"
                    placeholder=""
                    value={email}
                    onChange={(e) => setEmail(e.target.value)}
                  />
                </div>
                <div className="mb-6 w-full">
                  <label className="block text-gray-100 text-lg font-medium
mb-2" htmlFor="password">
                    Password
                  </label>
                  <div className='flex h-fit gap-4 items-center'>
                    <input
                      className="shadow text-lg appearance-none border
rounded w-full py-2 px-3 text-gray-700 leading-tight focus:outline-none
focus:shadow-outline"
                      id="password"
                      type={hide.type}
                      placeholder=""
                      value={password}
                      onChange={(e) => setPassword(e.target.value)}
                    />
                    <button type='button' onClick={handlePass} className='px-4
py-2 rounded-xl bg-blue-500 font-medium text-white'><i
className={hide.name}></i></button>
                  </div>
                </div>
                <div className="flex items-center justify-between w-full">
                  <button
                    className="bg-blue-500 hover:bg-blue-700 text-white font-
```

```
medium w-full py-2 px-4 rounded focus:outline-none focus:shadow-outline"
                    type="submit"
                >
                    Submit
                </button>
            </div>
            <div className='flex justify-between w-full mt-4'><p>Have an
account ?</p> <Link className='text-blue-700 border-b px-1 border-blue-
700' to='/login'>Login</Link></div>
        </form>
    </div>
  );
};

export default Signup;
```




```
import React, { useState ,useEffect} from 'react';
import { Link, useNavigate } from 'react-router-dom'; // Import useNavigate here
import { ToastContainer, toast } from 'react-toastify';
import 'react-toastify/dist/ReactToastify.css';

const Login = ({ setLogin, setName, setemail, setproduct }) => {
  const navigate = useNavigate(); // Move useNavigate inside the Login component
  const notify = () => toast("Login Sucessful");
  const [email, setEmail] = useState('');
  const [password, setPassword] = useState('');
  const [hide, setHide] = useState({
```

```
      name: 'bx bx-show',
      type: 'password'
    });


    const handlePass = () => {
      hide.name === 'bx bx-show' ? setHide({ name: 'bx bx-hide', type: 'text' }) :
setHide({ name: 'bx bx-show', type: 'password' });
    };

    const handleSubmit = async (e) => {
      e.preventDefault();
      try {
        const response = await fetch('http://localhost:8080/login', {
          method: 'POST',
          headers: {
            'Content-Type': 'application/json'
          },
          body: JSON.stringify({
            email,
            password
          })
        });

        if (response.ok) {
          // Login successful, handle success
          toast.success('Login  Successfully');
          setLogin(true);
          setTimeout(() => {
            navigate("/")
          }, 1500);

          try {
            // Convert response to JSON
            const data = await response.json();
            // Destructure data object
            const { email, id, name,
password,productAbout,productName,productPrice,productUrl } = data;
              setemail(email)
              setName(name)
              setproduct({productAbout,productPrice,productName,productUrl})
          } catch (error) {
            toast.error('Error parsing response data');
```

```
            }
        } else {
            toast.error('Incorrect details')
        }
    } catch (error) {

        toast.error('Error log in user');
    }

};

return (
    <div className=" loginbg flex justify-center items-center h-screen">
        <form onSubmit={handleSubmit} className=" glass bxshadow text-white
shadow-md rounded w-[40%] h-[70%] flex flex-col justify-center items-start p-16">
            <h1 className=' self-center pb-4 mb-6 border-b px-5 headingfont text-3xl
text-white'>Login</h1>
            <div className="mb-4 w-full">
                <label className="block text-gray-200 text-lg font-bold mb-2
headingfont" htmlFor="username">
                    Email
                </label>
                <input
                    className="shadow appearance-none border rounded w-full py-2 px-
3 text-gray-700 leading-tight focus:outline-none focus:shadow-outline"
                    id="username"
                    type="text"
                    placeholder=""
                    value={email.toLowerCase()}
                    onChange={(e) => setEmail(e.target.value)}
                />
            </div>
            <div className="mb-6 w-full">
                <label className="block text-gray-100 text-lg font-bold mb-2
headingfont" htmlFor="password">
                    Password
                </label>
                <div className=' flex h-fit gap-4  items-center'>
                <input
                    className="shadow text-lg appearance-none border rounded w-full
py-2 px-3 text-gray-700  leading-tight focus:outline-none focus:shadow-outline"
                    id="password"
                    type={hide.type}
```

```
                placeholder=""
                value={password}
                onChange={(e) => setPassword(e.target.value)}
              />
              <button type="button" onClick={handlePass} className=' px-4 py-2
rounded-xl bg-blue-500 font-medium text-white'><i
className={hide.name}></i></button>

            </div>
          </div>
          <div className="flex items-center justify-between w-full">
            <button
              className="bg-blue-500 hover:bg-blue-700 headingfont text-white
font-bold w-full py-2 px-4 rounded focus:outline-none focus:shadow-outline"
              type="submit"
            >
              Sign In
            </button>
            <ToastContainer/>
          </div>
            <div className=' flex justify-between w-full mt-4'><Link
className='border-blue-700 border-b px-1 text-blue-700' to='/forgetpass'> Forget
Password ?</Link> <Link className=' text-blue-700 border-b px-1 border-blue-700'
to='/signup'>Create an account</Link></div>
        </form>
      </div>
  );
};

export default Login;
```

SYSTEM TESTING

The testing process focuses on the Logical intervals of the software ensuring that all statements have been tested and on functional interval s conducting tests to uncover errors and ensure that defined input will produce actual results that agree with the required results, Program level testing. modules level testing integrated and carried out.

Testing Methods There are two major types of testing they are

- **White Box Testing**
- **Black Box Testing**

White Box Testing :

White box sometimes called Glass box testing a test case design uses the control structure of the procedural design to drive test case.

Black box Testing :

Black box testing focuses on the functional requirements of the software. This s black box testing enables the software engineering to derive a set of input conditions that will fully exercise all functional requirements for a program Black box testing is not an alternative to white box testing rather it is complementary approach that is likely to uncover a different class of errors that white box methods like.

- Interface errors
- Performance in data structure
- Performance errors
- Initializing and termination errors

Unit testing :

Unit testing is software verification and validation method in which a programmer tests if individual units of source code are fit for use. A unit is the smallest testable part of an application. In procedural programming a unit may be an individual function or procedure. Ideally, each test case s independent from the others: substitutes like method stubs, objects, fakes can be used to assist testing a module in isolation.

Integration Testing:

This testing is sometimes called Integration and Testing Integration testing the phase in software testing in which individual software modules are combined and tested as a group. It occurs after unit testing and before system testing, Integration testing takes as its input modules that have been unit tested, groups them in larger aggregates, applies tests defined in an integration test pan to those aggregates and delivers as its output the integrated system ready for system testing

Validation Testing :

Validation Testing can be defined in many ways, but a simple definition is that validation succeeds when the software functions in a manner that can reasonably expected by a customer. After validation test has been conducted, one of the following two possible conditions exist. The functions or performance characteristics confirm to specification and are accepted. In the administrator and login modules, all the fields must be filled.

User Acceptance Testing :

User acceptance of a system is a key factor of any system. The system under consideration is tested tor the acceptance by constantly keeping in touch with the prospective system users at the same time of developing and marketing changes when required.

FUTURE SCOPE OF THE PROJECT

The future of wedding plan management is likely to see a focus on technology and personalization, making the process more efficient, data-driven, and tailored to individual couples. Here are some trends to look for:

AI-powered planning: Imagine an AI assistant that walks couples through the planning process, suggesting vendors, creating budgets, and even recommending tasks based on their preferences and timeline.

Hyper-personalization: Wedding planning tools will leverage data to personalize the experience. This could include suggesting vendors based on past client preferences and budget, or recommending venues with accessibility features for specific guest needs.

Virtual reality experiences: Couples could use VR technology to virtually tour venues, see how their chosen decorations will look in a specific space, or even try on different wedding attire virtually.

Sustainable planning tools: There will be a growing demand for tools that help couples plan eco-friendly weddings. This could include recommending sustainable vendors, providing tips on reducing waste, and calculating the carbon footprint of the event.

Focus on guest experience: Wedding planning tools will incorporate features that enhance the guest experience. Imagine interactive RSVPs with dietary restrictions, digital invitations with embedded maps and parking information, or even live-streaming options for remote guests.

Subscription-based planning services: Subscription models could offer access to a wider range of tools, expert advice, and exclusive vendor discounts.

Integration with the wedding industry: Wedding plan management tools will integrate seamlessly with other wedding industry services, such as online invitations, registries, and payment platforms.

By embracing these trends, wedding plan management platforms can become even more valuable for couples, helping them create a stress-free and unforgettable wedding experience.

CONCLUSION

Wedding plan management websites have revolutionized the way couples plan and execute their weddings, providing a comprehensive suite of tools that streamline every aspect of the planning process. From budget tracking and guest list management to vendor coordination and timeline planning, these platforms offer essential features that ensure a seamless and stress-free experience. As technology continues to advance, the future holds exciting possibilities for further enhancing these services with personalized recommendations, virtual reality tours, and integrated e-commerce options. By continually adapting to the evolving needs of users and leveraging the latest technological innovations, wedding plan management websites will remain invaluable resources, helping couples bring their dream weddings to life with ease and efficiency.

REFERENCES AND BIBLIOGRAPH

- www.google.com
- www.Wikipedia.com
- https://www.javatpoint.com/
- https://spring.io/projects/spring-boot