# Software Project Development on

|  |
| --- |
| “MANAGE NSS APP” |

Submitted to: **Smt K.B Parekh college of Computer Science-Mahuva**

## "Affiliated with the prestigious Maharaja Krishnakumarsinhji Bhavnagar University"https://www.mkbhavuni.edu.in/mkbhavuniweb/images/atpl-mkbu.jpg

In partial fulfillment of award of degree of

BACHELOR

OF

COMPUTER APPLICATIONS

Submitted by:

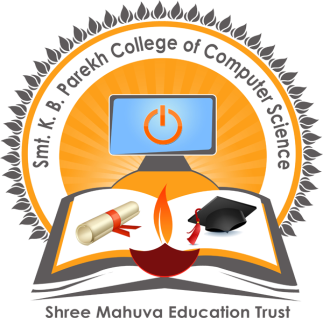
***Makavana Shailesh***

Guided by:

**Name and signature of project guide:** **Signature of Principal:**

**Smt K.B Parekh College of Computer Science Mahuva-364290**

Affiliated to Maharaja Krishnakumarsinji Bhavnagar University

****

***Acknowledgement***

It is our great pleasure to acknowledge the contribution and assistance of a few individuals in this project.

This project is prepared in the partial fulfillment of the requirement for the degree of Bachelor in Computer Application (BCA). 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. On doing this undergraduate project we have been fortunate to have help, support and encouragement from many people we would like to acknowledge them for their cooperation.

Our heartfelt thanks go to, thanks for sharing his valuable time with us to evaluate our work and provide professional suggestion with us.

This project has been a wonderful experience where we have learnt and experienced many beneficial things.

Finally we would like to thank everyone who directly or indirectly helped us in the project.

**INDEX**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| NO | INDEX | | | Page |
| 1 | Introduction | | | 4 |
|  |  | | Background | 5 |
|  |  | | Purpose | 6 |
|  |  | | Objectives | 7 |
|  |  | | Scope | 8 |
| 2 | Requirement and Analysis | | |  |
|  |  | | Problem definition | 9 |
|  |  | | Requirement specification | 10 |
|  |  | | Hardware software Requirement | 12 |
|  |  | | Planning and scheduling | 13 |
| 3 | System design | | |  |
|  |  | | Over All System Design Using Designing Tools | 16 |
|  |  | | Data Dictionary | 25 |
|  |  | | Input/output Design | 30 |
| 4 | Testing and Implementation | | |  |
|  |  | Testing Approach | | 40 |
|  |  | Test cases | | 44 |
|  |  | Implementation Approaches | | 45 |
| 5 | Conclusion | | |  |
|  |  | Conclusion | | 48 |
|  |  | Limitations of System | | 49 |
|  |  | Future Scope of System | | 49 |
|  |  | Bibliography | | 50 |

CHAPTER: 1 INTRODUCTION

* This project is aimed at developing a Android application that depicts online nss management system.
* Using This Application manage every things of NSS like :
* Student manages
* Event manages
* Student registration
* Nss store items
* Event Images
* Manage all registration
* The application is reduced as much as possible to avoid errors while entering the data. It also provide error message while entering invalid data.
* Using this app easy to maintain all student record and events data.
* Create event for student and volunteers to easily get information of any event and easily registration any event without any fix time.
  + Project Background:
* Idea About How online NSS App is work:
* The online NSS app work on the basis of take detail of Student or gest users, and register any event to manage student and volunteer registration
* For make a system computerized.
* To improve the system performance and easy to search a any event or check past event data.
* In old paper-based system all the records are stored in paper but after some times that are difficult to place and store, and some time letter paper can be damaged and we lost a record at that time computerized system become useful.
* In paper-based system paper also is faulted than admin also lost the record.
  + Purpose:
* Main purpose of **NSS** app is to save time, save money and easily manage registration.
* Using this NSS app a student any time register in any event no any time limit, no any fix time,
* Using this app easily manage all student in any event resisted,
* In this app the admin can manage all event, images, student and NSS store items
* Using this app easily check how many items available in NSS store for example T-Shirt is 200 , it direct check in this app .
  + Objectives:
* The objective of **“NSS application”** is to develop an easily manage all student , store and event .
* This application user interactive an easily manage user profile.
* In this app the student any time register in event,
* The software will help in easy maintaining and updating event in the app for the administrator.
* Using this app the administrator is manage all student registration accept or reject unauthorize any user.
* This app is maintain the nss store , any time the administrator check nss items
  + Scope:
* This app is reduce the paper work for manage student registration , check who is register in any event
* And also generates the various reports for analysis. Main concept of the project is to enter transaction reports and to maintain customer records.
* Hence, this software can be used in any users to maintain their record easily.
* This app help to maintain images of any particular image and event users
* Online Mobile Shop has as great future scope.

CHAPTER: 2 REQUIREMENT AND ANALYSIS

* Problem Definition
* So much Paper work.
* Process is much time consuming.
* Extra expense in paper work.
* Manually student registration.
* Time bearer.
* Accuracy and consistency in manual system is less.
* Personal delay.
* In manual system it is tedious task to search a particular record. later after.
* Student attendance.
* Manage NSS store items.
* Requirement specification
* As per the System Requirements it contains two (4) Modules:
* ***Functional Requirement of all Modules:***

1. Admin:

Admin has all rights in the software including updating the status of his site. The admin has authority to Add, Update, and Delete.

1. User:

The User can only view Event, and register in any event.

View all old event images.

* ***Non-Functional Requirement:***

1. Supportability:

The system will display meaningful error messages to the user of the system that will help user to resolve the problem. For that we have been done optimized coding using the key events of **EXPO** react native to achieve client side validation.

1. Security:

The system will be secure as each user of the system has its own access permission, and access permission can only be changed by the admin user.

The system will be secure as backup of all the data is being carried out regularly.

For better database security we use MY SQL.

1. Look and Feel

The look and feel of the NSS is based on various types of events. User can get the required description when he/she clicks to the event view this event.

* Hardware software Requirement
* ***Hardware Requirement:***
* Pentium 4, 2.4 GHz Processor
* 80GB HDD
* 512MB RAM
* Android device
* ***Software Requirement***
* NODE JS : 20.18.1
* My SQL
* Expo go app (for run app ANDROID)
* Planning and scheduling

Different amount of time may be required for each stage in the project cycle, depending on the particulars of the key aspect of the project cycle seem to recur during development process. The information obtained during the requirement gathering of pre-development phase provides the impetus for the requirement analysis and the information is further used in the design phase.

|  |  |  |
| --- | --- | --- |
| **Task** | **Start-Finish** | **Duration** |
| Analysis | 1/1/25 to 4/1/25 | 4 Days |
| Design | 8/1/25 to 14/1/25 | 11 Days |
| Coding | 18/1/25 to 23/2/25 | 38 Days |
| Testing | 6/2/25 to 11/2/25 | 6 Days |
| Implementation | 15/2/25 to 20/2/25 | 5 Days |
| Updating | 21/2/25 to 27/2/25 | 3 Days |
| Documentation | 1/3/25 to 12/3/25 | 13 Days |

**(Diagram 1:- Planning and Scheduling)**

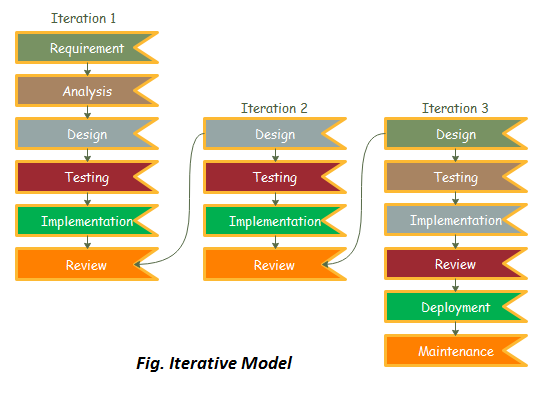
## The above schedule specifies the estimated time that will be required in various software development phases, considering all situational factors. Team members are technically ready accepting few days training on to get the Technology Awareness. Thus, according to calculation, it is feasible to build such solution in time. “The schedule will be revised at the end of each phase and updated as required”.

* ***Selecting the Iterative Model as Project Development Strategy***

Iterative Model is way of breaking down the software development of a large application into smaller parts. In Iterative Model, features code is designed, developed and tested in repeated cycles. With each iteration, additional features can be designed, developed and tested until there is a fully functional software application ready to be deployed to customer.

Typically, iterative model is used in conjunction with incremental development in which a longer software development cycle is split into smaller segments that build upon each other.

Diagram 2 (Iterative Model)



* ***Iterative Model Advantages:***
* Generates working software quickly and early during the software life cycle.
* More flexible – less costly to change scope and requirements.
* Easier to test and debug during a smaller iteration.
* Easier to manage risk because risky pieces are identified and handled during its iteration.
* Each iteration is an easily managed milestone
* ***Iterative Model Disadvantages:***
* Each phase of an iteration is rigid and do not overlap each other.
* Problems may arise pertaining to system architecture because not all requirements are gathered up front for the entire software life cycle.

CHAPTER: 3 SYSTEM DESIGN

* **Over All System Design Using Designing Tools**

The Purpose of Design Phase is to plan a solution for problem specified by the requirements. System Design aims to identify the modules that should be in the system, the specification of those modules and how they interact with other to produce the results. The goal of the design process is to produce a model that can be used later to build that system. The produced model is called design of the system.

System Design is the process of defining the architecture, components, modules, interfaces and data for a system to satisfy specified requirements.

Normally, the design proceeds in 2 stages:

* Physical Design
* Database Design

1. **Physical Design**

The Physical Design is a graphical representation of a system showing the system’s internal and external entities and the flow of data into and out of these entities. An internal entity is an entity within the system that transforms data.

To represent the Physical Design of the system, we use diagrams like Data Flow Diagrams, E-R Diagrams, Use Case Diagrams, etc.

* **Data Flow Diagrams**

The Data Flow Diagrams (DFD) is a graphical representation of the flow of data through an information system. Data Flow Diagrams are used by systems analysis to design information processing systems but also a way to model whole organization. The main merit of DFD is that it can provide an overview of what data a system would processes. What transformations of data are done, what data are stored and which stored data is used, and where the result is flow.

* **Standard Symbols used in DFD:**
* **Processes:**

Represented By Circles or Ovals, Processes in A DFD Represent the Functions or Activities That Transform Input Data into Output Data. Each Process Performs a Specific Function or Computation Within the System.

* **Dataflow:**

Represented By Arrows, Data Flows In A DFD Depict The Movement of Data Between Processes, Data Stores, And External Entities. They Show The Flow of Information Within The System And Indicate How Data Is Processed And Transmitted.

* **Data Store:**

Represented By Rectangles, Data Stores In A DFD Represent Repositories Or Storage Locations Where Data Is Retained Within The System. They Can Include Databases, Files, Or Other Data Storage Mechanisms.

* **Processes:**

Represented By Squares Or Rectangles With Rounded Corners, External Entities In A Dfd Represent Sources Or Destinations of Data That Interact With The System But Are External To It. They Can Include Users, Other Systems, Or External Organizations.

## Level 0 (Context Diagram):

* Also Known As The Context Diagram, Level 0 Provides An Overview Of The Entire System, Showing The System As A Single Process Or Function Interacting With External Entities.
* It Represents The Highest Level Of Abstraction And Focuses On The Interactions Between The System And Its Environment, Without Delving Into The Internal Workings Of The System.

Sign in & Sign up

HOME

Events

Event Register

View Past Events

Explore

Store

* **1st level DFD Admin:**

The 1st level DFD’s aim is give to detail overview of the full system. The 1st level DFD of nss management system is given below:

Add / edit item

request

EVENTS

Add / edit events

Response

request

USERS

Reply

Reply

Response

ADMIN

USERS

LOGIN

request

Reply

Response

MANAGE EVENTS

MANAGE USERS

request

Edit imimage

response

IMAGES

edit

replay

STORE

Reqest

responce

MANEGE REGISTRATION REQUEST

REGISTRATRATION

response

RESPONCE

* **User 1st level DFD USERS**

USER

users

Sign in

users

Sign up

Edit Profile

users

registration

**(Diagram 6:- 1st Level User DFD)**

* **User 2st level DFD**

users

user

events

images

registration

users

**(Diagram 7:- 2nd Level User DFD)**

* **E-R Diagram (Entity-Relationship Diagram)**

Entity-Relationship Diagram is a graphical representation of entities and their relationship to each other. It describes how data is related to each other. An entity is a piece of data- an object or a concept about which data is stored. A relationship is how the data is shared between entities.

In E-R Diagram, there are 3 main components:

|  |  |  |
| --- | --- | --- |
| **Symbol** | **Name** | **Description** |
|  | **Entity** | An Entity can be any object, place, person or anything. |
|  | **Attribute** | An Attribute describes a property or characteristics of an entity |
|  | **Relationship** | A Relationship describes relation between entities. |

**(Diagram 8:- E-R Diagram Symbol Representation)**

ID

NAME

PASSWORD

DESCRIPTION

IMAGE

EMAIL

NAME

HAS

EVENTS

ROLE

USERS

HAS

PHONE

Image

REGISTRATION

NSS\_store

* Data dictionary

## Database Design / Data Structure Design

Various tables used in the system to manage the data are as follow:

1. USERS
2. EVENTS
3. REGISTRATION
4. NSS\_STORE
5. IMAGES
6. USERS :

Table Name: USERS

Primary key: id

Description: this table used to manage all user available in nss system user like : admin , student

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field Name** | **Data Type** | **Length** | **Key** | **Description** |
| **Id** | Int | - | Primary Key, Auto Increment | Unique user is |
| **name** | Varchar | 50 | NOT NULL | User name |
| **email** | VARCHAR | 100 | Unique, NOT NULL | Email for sign in signup |
| **password** | VARCHAR | 100 | NOT NULL | Password sign in |
| **phone** | VARCHAR | 20 | **-** | number |
| **role** | ENUM('1','a') | - | DEFAULT '1' | Role for access |
| **img** | VARCHAR | 255 | **-** | Image URL |

* Following all field and datatype

1. Events Table :

Table Name: events

Primary key: id

Description: This table is for manage all events available in nss app and manage upcoming events

* Following table is represent full table view

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field Name** | **Data Type** | **Length** | **Key** | **Description** |
| **Id** | Int | - | Primary Key, Auto Increment | Unique all events id |
| **name** | Varchar | 255 | NOT NULL | Event name |
| **description** | TEXT | - | NULL | Email for sign in signup |
| **location** | VARCHAR | 255 | NULL | Give Event Location |
| **start\_time** | DATETIME | - | NOT NULL | Event Starting date |
| **end\_time** | DATETIME | - | NOT NULL | Event Ending date |
| **numOFUser** | INT | - | **0** | How Many Student Are Participate |
| **image** | VARCHAR | 255 | NULL | Event image url main image |
| **Create\_by** | INT | - | Users(id) | Foreing key from users |

1. images :

Table name: images

Primary key: id

Description: This table shows the all images of all events

* Following table is represent full table view

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field Name** | **Data Type** | **Length** | **Key** | **Description** |
| **Id** | Int | - | Primary Key, Auto Increment | Unique all every iamge id |
| **imageurl** | Varchar | 255 | NOT NULL | Event name |
| **E\_id** | INT | - | Events(id) | Foreing key from events |

1. NSS STORE :

Table name: nss\_store

Primary key: id

Description: This table shows the all images of all items

* Following table is represent full table view

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field Name** | **Data Type** | **Length** | **Key** | **Description** |
| **Id** | Int | - | Primary Key, Auto Increment | Unique all every iamge id |
| **item\_name** | Varchar | 255 | NOT NULL | Items name |
| **number\_of\_items** | INT | - | NOT NULL | How many items valable |

1. Event registration :

Table name: registration

Primary key: id

Description: This table shows the all registration users in any event

* Following table is represent full table view

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field Name** | **Data Type** | **Length** | **Key** | **Description** |
| **Id** | Int | - | Primary Key, Auto Increment | Unique all every registration id |
| **user\_id** | INT | - | NOT NULL, usres(id) | Who Is Register |
| **event\_id** | INT | - | NOT NULL,  Events(id) | Which Event In Registers Student |
| **registration\_date** | TIMESTAMP | - | CURRENT\_TIMESTAMP | Registration date |
| **status** | ENUM('pending', 'confirmed', 'cancelled') | - | - | Registration is accept ot not |

## Input/output Design

* ***User Screen***

|  |  |
| --- | --- |
| Splash Screen | Loading Screen(auth) |
|  |  |

|  |  |
| --- | --- |
| Sign In Screen | Reset Password |
|  |  |

|  |  |
| --- | --- |
| Sign Up Screen | Home Screen |
|  |  |

|  |  |
| --- | --- |
| Full Event Screen | All Events Screen |
|  |  |

|  |  |
| --- | --- |
| Explore Screen | Profile Screen |
|  |  |

|  |  |
| --- | --- |
| Sign In Screen |  |
|  |  |

|  |  |  |
| --- | --- | --- |
| Home | Users | Events |
|  |  |  |

* Admin Screens

|  |  |  |
| --- | --- | --- |
| Images | Store Items | Registration Event |
|  |  |  |

|  |  |  |
| --- | --- | --- |
| Update Gallery | Create New User | Update User |
|  |  |  |

|  |  |  |
| --- | --- | --- |
| Create Event | Update Event | Admin View User |
|  |  |  |

CHAPTER: 4 TESTING AND IMPLEMENTATION

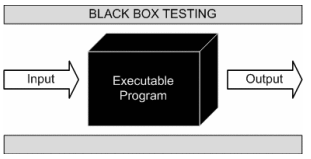
* 1. Testing Approaches used

The testing approach ensures software quality by focusing on **quality assurance, verification, and validation**. The testing process investigates the system’s performance, functionality, and security to provide insights into its quality. It also helps in identifying software bugs and errors. Through this approach, we ensure that **students can register, log in, and update their profiles without issues**, and **admins can efficiently manage student records**.

Various **testing techniques** were used to validate the system, including:

**Black Box Testing:**

In **Black Box Testing,** a tester doesn’t have any information about the internal working of the Software System. Black Box Testing is a high level of testing that focuses on the behavior of the Software. It involves testing from an external or end-user perspective. Black Box Testing can be applied to virtually every level of software testing: Unit, Integration, System, and acceptance.



## Advantages of Black box Testing:

* Tests are done from a user’s point of view and will help in exposing discrepancies in the specifications.
* Tester need not know programming languages or how the software has been implemented.

**2 . White Box Testing**

White-Box Testing is a Testing technique which checks the internal functioning of the system. In this method, Testing is based on coverage of code statements, branches, paths or conditions. White-Box Testing is considered as low-level Testing. It is also called glass box, transparent box, clear box or code base testing. The White-Box Testing method assumes that the path of the logic in a unit or program is known.

**White box testing techniques:**

1. Statement Coverage – This technique is aimed at exercising all programming statements with minimal tests.
2. Branch Coverage – This technique is running a series of tests to ensure that all branches are tested at least once.
3. Path coverage – This technique corresponds to testing all possible paths which means that each statement and branches is covered.

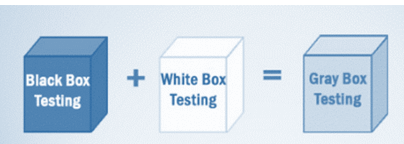
**Advantages of white box Testing:**

1. Forces test developer to reason carefully about implementation.
2. Reveals errors in “hidden” code.

**3. Gray Box Testing**

**Gray Box Testing** is a software testing method which is a combination of **Black Box Testing** method and **White Box Testing** method. In Black Box Testing**,** the internal structure of the item being tested is unknown to the tester and in White Box Testing the internal structure is known. In Gray Box Testing, the internal structure is known. This involves having access to internal data structures and algorithms for purposes of designing the test cases, but testing at the user, or black-box level.

Gray Box Testing is named so because the software program, in the eyes of the tester is like a gray **/** semi-transparent box**;** inside which one can partially see.



**Advantages of Gray Box Testing include:**

* Works well for large code segments.
* Testers are not required to know programming language or methods for testing the application, which makes this testing mostly unbiased and non-intrusive.
* Access to programming code is not essential.
* Provides clearly defined roles for users and developers during testing. Testing is based on the perspective of the user, rather than the designer.

4.2 **Test Cases**

A test case is a set of conditions or variables under which a tester or developer will determine whether a system under test satisfies the requirements or works correctly or not.

The process of developing test cases can also help in finding problems in the requirements or design of an application.

Test strategy we have done with module testing using gray box testing.

* 1. **Implementation Approaches:**

There are a variety of options that a project manager could consider when implementing a solution. There are Advantages and Dis-Advantages to each type, and the choice usually depends on the client organizational setup and the complexity of the solution to be implemented.

For example, an international client with multiple offices needs to upgrade a certain e-mail system in all offices by a certain “go-live” date. In such a scenario, a product manager is faced with huge logistical and technical challenges, and the implementation strategy is pivotal in deciding on the rollout.

These implementation choices available to a project manager are:

* Parallel Implementation
* Phased Implementation
* Crash Implementation
* **Parallel Implementation**

A parallel Implementation or approach implies that a new solution is implemented parallel to the current operating system in use. Those who are using the system will not see major downtime once it is implemented. The trick here is to implement the system.

Once the new solution is tested and up and running, it is “switched” on and the older version is “switched” off.

The advantages with a parallel implementation include:

1. Less disruption to the business
2. No loss of business if the new system suddenly fails.

* **Phased Implementation**

Sometimes trying to implement a solution all at once is not feasible because many clients have essential operations that run during normal working hours and cannot afford the luxury of having their entire operation close down for a lengthy period of time.

Often, clients have front office staffs that attend to these operations (such as Call centers, Help Desks, etc…), and they work in 24-hour shifts. This is why many clients approve of a phased implementation approach, and the project team must ensure that the phased implementation is possible. This approach involves implementing the solution to a certain amount of users and then rolling them onto the new solution, while the rest of the users are rolled out in a similar fashion until the entire solution is rolled out within the client environment.

The phase approach works well because:

1. There is minimal disruption to the clients operation

2. Problems are resolved quickly.

The phased approach could also be used if there is more than one department. The project manager could decide that implementing the solution in one department at a time could be more reliable than trying to roll out all departments at the same time.

* **Crash Implementation**

Careful planning needs to take place when considering a crash (also known as full-blown) implementation. It takes an incredible amount of planning to ensure no problems arise. In fact, with this type of implementation, the necessary contingencies need to be prepared and reviewed well in advance of the actual implementation, in order to minimize any potential failure.

The necessary IT support staffs also need to be available on the chosen implementation period. A full-blown implementation should be scheduled to take place over a slow period, such as a Holiday or Weekend.

CHAPTER: 5 CONCLUSIONS

# Conclusion

The NSS Management System app, built using React Native with Expo for the frontend and Node.js with MySQL for the backend, provides a seamless and efficient platform for managing NSS activities. This system enhances user engagement by enabling easy event registration, profile management, and event tracking. For administrators, it offers robust tools to manage events, registrations, users, and an image gallery, ensuring smooth coordination within the NSS community.

With a user-friendly interface and a structured database, this app improves accessibility and efficiency for both users and admins. By leveraging modern technologies, it simplifies the management process, fosters greater participation in NSS activities, and ensures an organized digital experience.

* **Limitations**

1. **Internet Dependency** – The app requires a stable internet connection for user authentication, event registration, and data retrieval, limiting its usability in offline scenarios.
2. **Platform Restrictions** – Currently designed for Android using Expo, the app may require additional configurations for full iOS compatibility.
3. **Limited Offline Support** – Users cannot register for events or update profiles without an active internet connection.
4. **Scalability Challenges** – As the number of users and events grows, performance optimization may be needed for smooth database operations.
5. **Image Storage Dependency** – If using Cloud-nary or similar services for image management, reliance on third-party platforms could pose challenges in terms of storage limits and costs.
6. **Security Considerations** – Without proper authentication and authorization mechanisms, user data and event details could be vulnerable to security threats.
7. **Device Compatibility** – Performance may vary across different Android devices, requiring extensive testing for optimal user experience.

* **Future scope**

1. **iOS Compatibility** – Extend the app to support iOS devices by ensuring compatibility with Apple’s ecosystem using Expo and React Native.
2. **Push Notifications** – Integrate push notifications to remind users about upcoming events, registration deadlines, and important updates.
3. **User Role Management** – Introduce different user roles with varied permissions, allowing more flexibility in event and profile management.
4. **QR Code-Based Event Check-In** – Implement QR code scanning for quick event attendance verification and tracking.
5. **Feedback and Ratings** – Enable users to rate and provide feedback on events, helping organizers improve future activities.
6. **Multi-Language Support** – Add support for multiple languages to enhance accessibility for users from different regions.