**Introduction:**

In this task we are going to list out the all the requirements and categorize them using MoSCoW prioritization. We also categorize them based on the functional and non-functional requirements. Using all the requirements Use Case diagram is also drawn and ER diagram has been also drawn. The initial class diagram for the project has been also made. The required UI design for the project has been made.

**Analysis**

**Functional Requirements:**

The functional requirement is describing the behavior of the system as it relates to the system's functionality. The non-functional requirement elaborates a performance characteristic of the system.

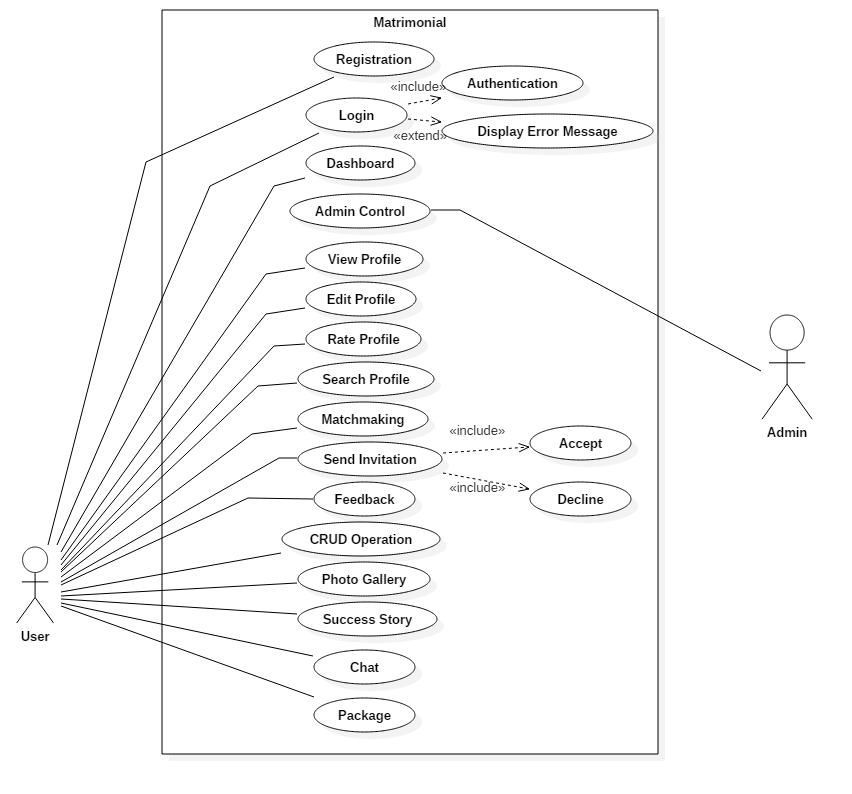
**Non-Functional Requirements:**

Non-functional requirements are sometimes defined in terms of metrics (i.e. something that can be measured about the system) to make them more tangible. Non-functional requirements may also describe aspects of the system that don't relate to its execution, but rather to its evolution over time (e.g. maintainability, extensibility, documentation, etc.).

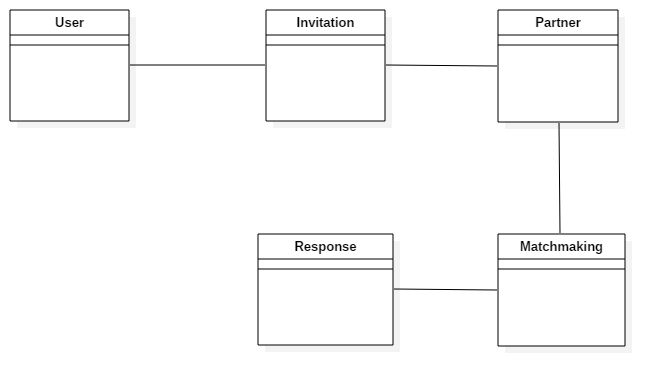
Requirements:

|  |  |  |
| --- | --- | --- |
|  | Requirements | MoSCoW |
| R1(F) | Registration | M |
| R2(F) | Login | M |
| R3(F) | Dashboard | M |
| R4(F) | Admin Control | M |
| R5(F) | View Profile | S |
| R6(F) | Edit Profile | S |
| R7(F) | Rate Profile | S |
| R8(F) | Search Profile | S |
| R9(F) | Matchmaking | M |
| R10(F) | Send Invitation | S |
| R11(F) | Feedback | W |
| R12(F) | CRUD Operation | M |
| R13(F) | Photo Gallery | C |
| R14(F) | Success Story | C |
| R15(F) | User Friendly | M |
| R16(NF) | Security | M |
| R17(F) | Message | S |
| R18(NF) | Performance | W |
| R19(NF) | Maintainability | S |
| R20(NF) | Usability | S |

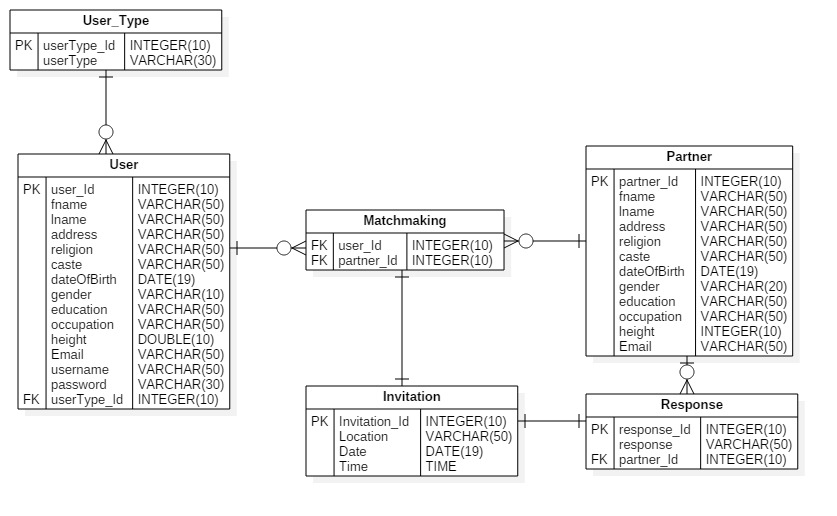
Use Case Diagram:



Initial Class Diagram:

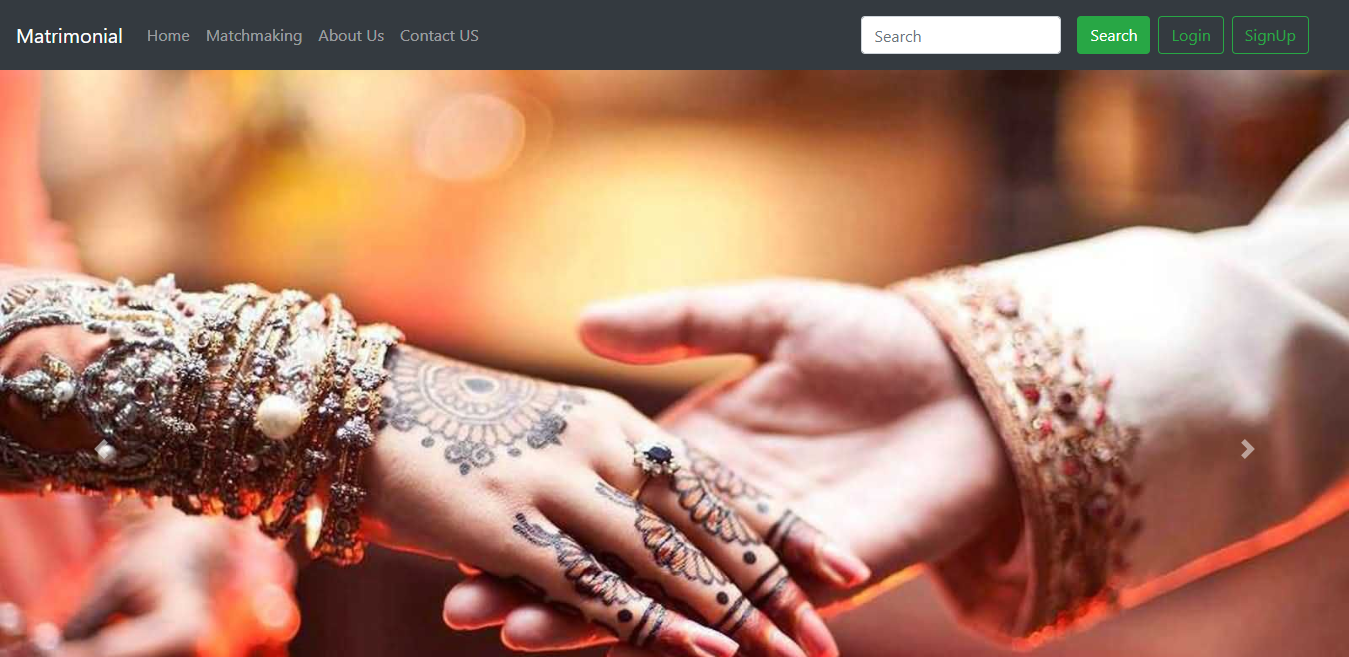


ER Diagram:

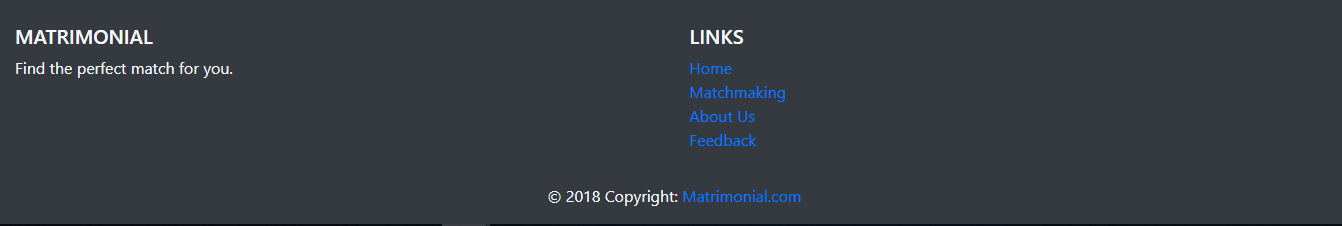


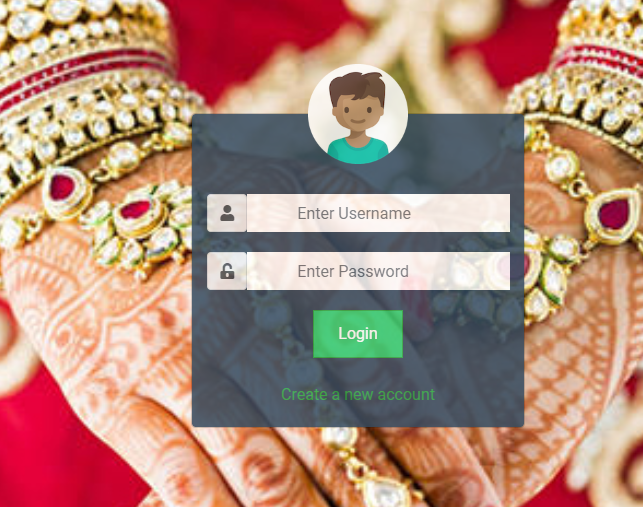
UI Design:

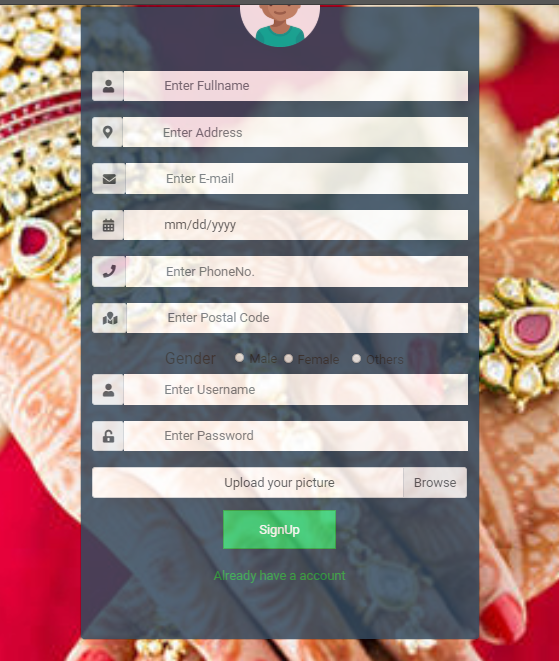
Homepage:

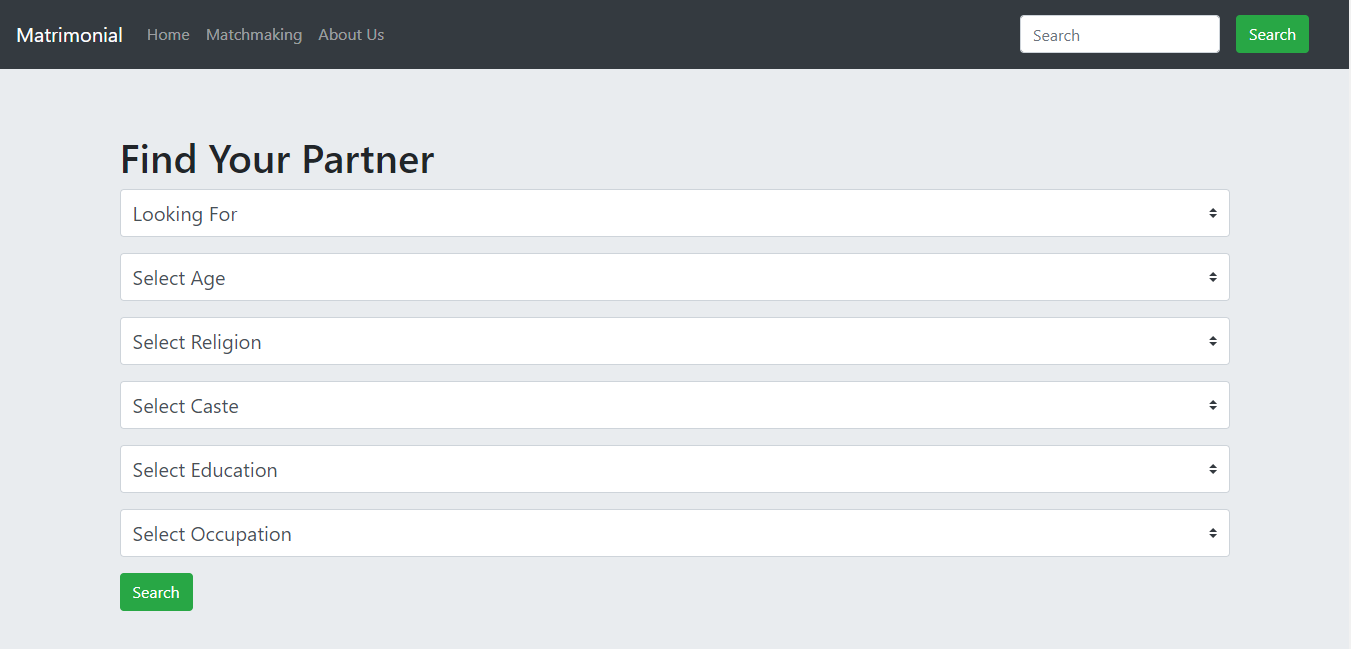


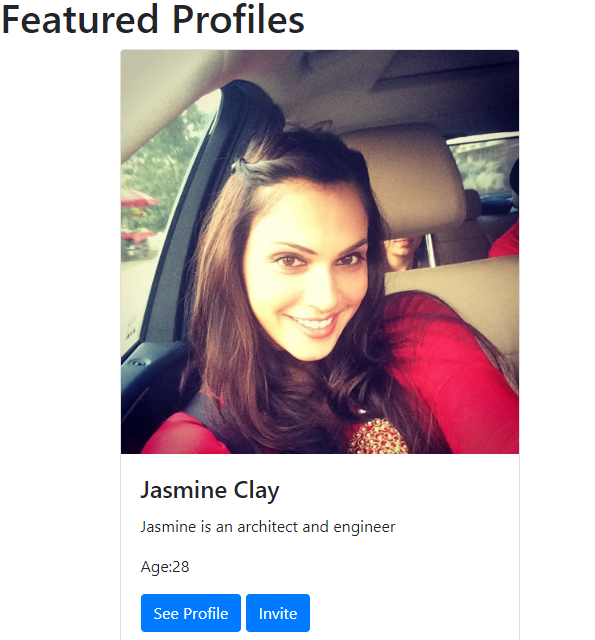
Footer:

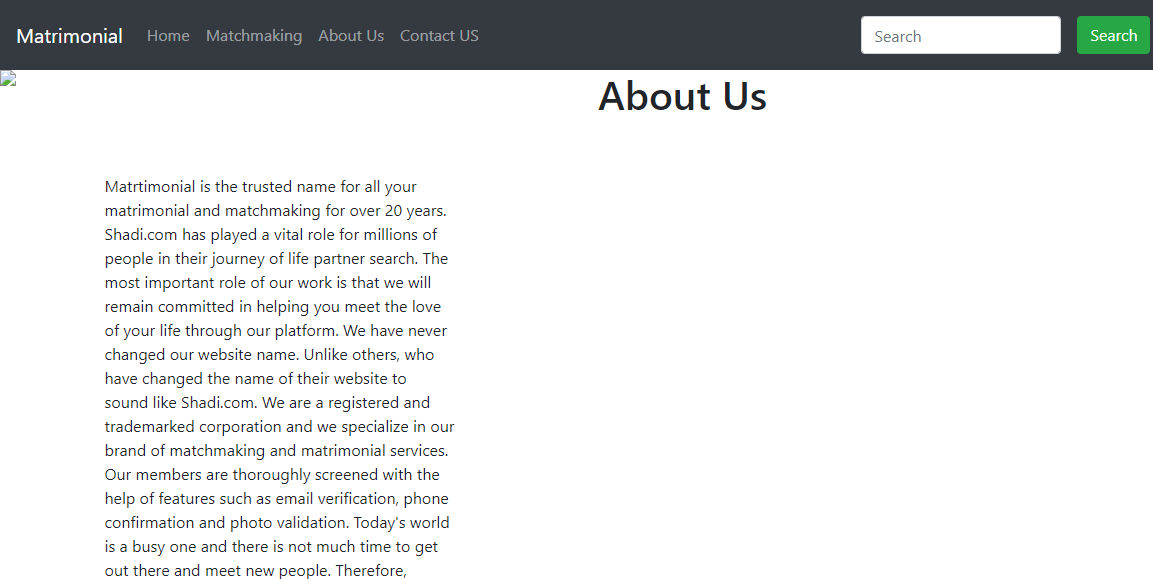


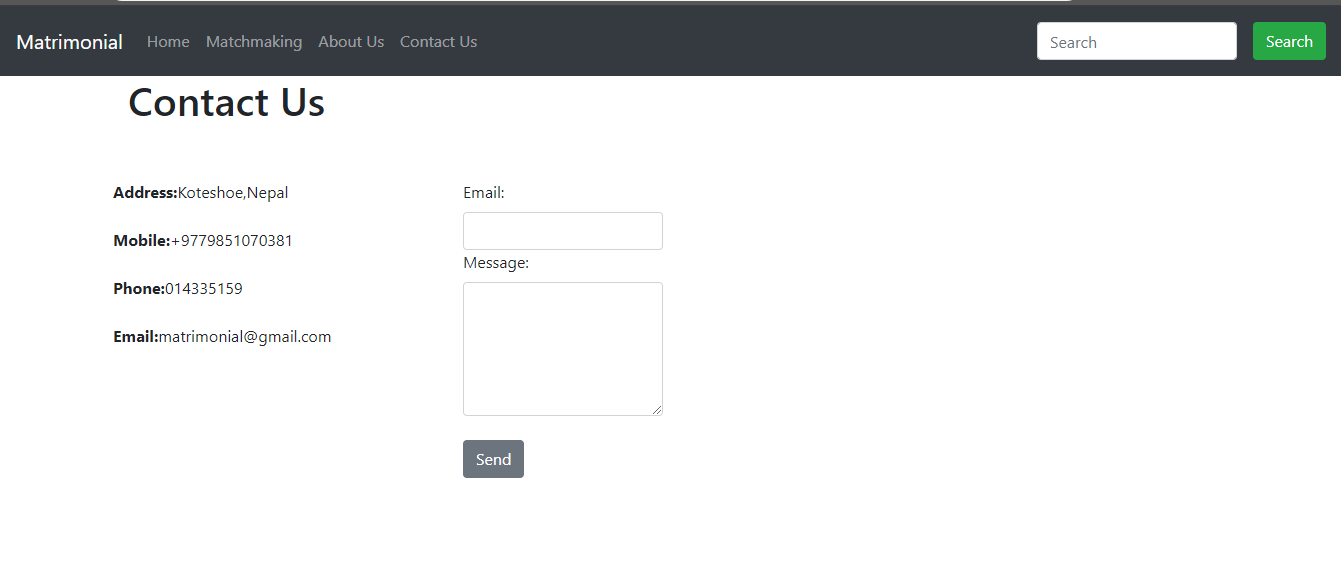
Login:

SignUp:

Finding Partner:

Featured Profiles:

About Us:

Contact Us:

Activity Diagram:

Activity diagram is another important diagram in UML to describe the dynamic aspects of the system.

Activity diagram is basically a flowchart to represent the flow from one activity to another activity. The activity can be described as an operation of the system.

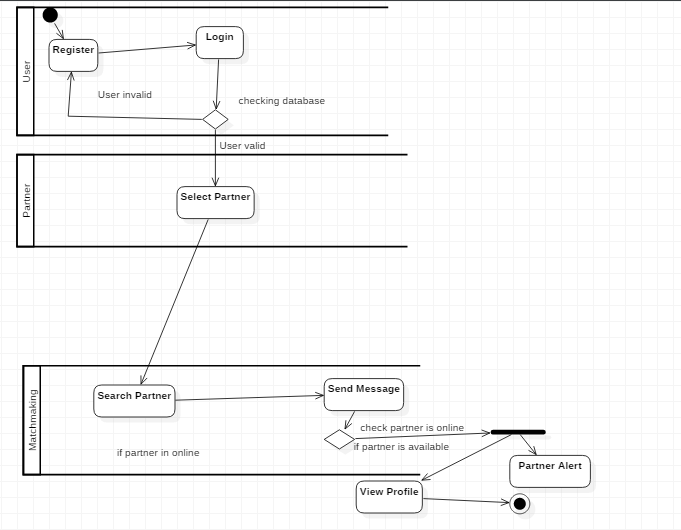
The control flow is drawn from one operation to another. This flow can be sequential, branched, or concurrent. Activity diagrams deal with all type of flow control by using different elements such as fork, join, etc.

Purpose of Activity Diagrams

The basic purposes of activity diagrams is similar to other four diagrams. It captures the dynamic behavior of the system. Other four diagrams are used to show the message flow from one object to another but activity diagram is used to show message flow from one activity to another.

Activity is a particular operation of the system. Activity diagrams are not only used for visualizing the dynamic nature of a system, but they are also used to construct the executable system by using forward and reverse engineering techniques. The only missing thing in the activity diagram is the message part.

It does not show any message flow from one activity to another. Activity diagram is sometimes considered as the flowchart. Although the diagrams look like a flowchart, they are not. It shows different flows such as parallel, branched, concurrent, and single.



Class Diagram:

Class diagram is a static diagram. It represents the static view of an application. Class diagram is not only used for visualizing, describing, and documenting different aspects of a system but also for constructing executable code of the software application.

Class diagram describes the attributes and operations of a class and also the constraints imposed on the system. The class diagrams are widely used in the modeling of object oriented systems because they are the only UML diagrams, which can be mapped directly with object-oriented languages.

Class diagram shows a collection of classes, interfaces, associations, collaborations, and constraints. It is also known as a structural diagram.

## Purpose of Class Diagrams

The purpose of class diagram is to model the static view of an application. Class diagrams are the only diagrams which can be directly mapped with object-oriented languages and thus widely used at the time of construction.

UML diagrams like activity diagram, sequence diagram can only give the sequence flow of the application, however class diagram is a bit different. It is the most popular UML diagram in the coder community.

The purpose of the class diagram can be summarized as −

* Analysis and design of the static view of an application.
* Describe responsibilities of a system.
* Base for component and deployment diagrams.
* Forward and reverse engineering.