CELEBREMOS

Software Requirements Specification

Version 6.0

12/01/2019

Hari Priya Jupally

Karun Bourishetty

Indra Reddy Mamidi

Vishal Reddy Pannala

Saikumar Nalivela

Poojitha Singam

Sairam Badisa

Submitted in partial fulfillment

Of the requirements of

CSIS 44-691 Graduate Directed Project II

# Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Description** | **Author** | **Comments** |
| 06/11/2019 | Version 1.0 | Poojitha Singam | Requirements Gathering |
| 06/12/2019 | Version 2.0 | Hari Priya Jupally | ER Diagram |
| 06/13/2019 | Version 3.0 | Indra Reddy | Prototype |
| 06/30/2019 | Version 4.0 | Sai Ram Badisa, Poojitha Singam,  Hari Priya Jupally | Technical Manual |
| 11/10/2019 | Version 5.0 | Indra Reddy | Functionalities |
| 12/01/2019 | Version 6.0 | Hari Priya Jupally  Sai Ram Badisa, | User Manual |

# Document Approval

The following Software Requirements Specification has been accepted and approved by the following:

|  |  |  |  |
| --- | --- | --- | --- |
| **Signature** | **Printed Name** | **Title** | **Date** |
| Zhengrui Qin | Dr. Zhengrui Qin | Client | 09/29/2019 |
| Vishal Reddy Pannala | Vishal Reddy Pannala | Program Lead Developer | 09/30/2018 |
| Saikumar Nalivela | Saikumar Nalivela | Programmer | 11/18/2018 |
| Indra Reddy Mamidi | Indra Reddy Mamidi | UI Developer | 12/02/2018 |
| Hari Priya Jupally | Hari Priya Jupally | Project Manager | 04/21/2019 |
| Sairam Badisa | Sairam Badisa | Tester |  |
| Karun Bourishetty | Karun Bourishetty | Team Lead |  |
| Poojitha Singam | Poojitha Singam | Database Administrator |  |

**Table of Contents**

|  |  |  |
| --- | --- | --- |
| **S.no** | **Contents** | **Page no.** |
| **1** | **Introduction**   * 1. Purpose   2. Scope   3. Definitions, Acronyms, and Abbreviations   4. References   5. Overview |  |
| **2** | **General Description**   * 1. Product Perspective   2. Product Functions   3. User Characteristics   4. General Constraints   5. Assumptions and Dependencies |  |
| **3** | **Specific Requirements**   * 1. External Interface Requirements      1. User Interfaces      2. Hardware Interfaces      3. Software Interfaces      4. Communications Interface   2. Functional Requirements   3. Use Cases   4. Class/Objects   5. Non-Functional Requirements   3.5.1. Performance  3.5.2. Reliability   * + 1. Availability     2. Security     3. Portability   1. Inverse Requirements   2. Design Constraints   3. Logical Database Requirements   4. Other Requirements   5. Prototypes (for complete project)   6. Use Case Diagrams |  |
| **4** | **Design**  4.1 ER diagram  4.2. GUI |  |
| **5** | **Analysis Models**  5.1. Data Flow Diagram  5.2. Sequence Diagram |  |
| **6** | **Technical Manual**  6.1 Login page  6.2 Registration  6.3 Create Events  6.4 Dashboard  6.5 Cards and Contacts  6.6 RSVP  6.7 Database Functionality  6.8 Messaging |  |
| **7** | **End User Manual**  7.1 Login page  7.2 Registration  7.3 Forgot Password  7.4 Create Events  7.5 Dashboard  7.6 Cards and Contacts  7.7 RSVP |  |

**UNIT 1: INTRODUCTION**

* 1. **Purpose**

The main purpose of the application is to create, modify and delete an online invitation system which helps the people to invite guests for their party to manage the number of people attending the event, creating polls to finalize the food menu and what costume they have to wear. This application helps the event organizers to plan the event successfully without any issues.

* 1. **Scope**

The scope of the project is to create an application that helps the event organizers to invite the guests to party which helps the organizers to plan for the place, food items, number of guests, what are the lists of items they need to bring while coming to the party and what costume they have to wear.

* 1. **Definitions, Acronyms, and Abbreviations**

RSVP - Répondez s'il vous plaît (Please respond)

ER - Entity Relationship

* 1. **References**

Retrieved from evite website:

<https://www.evite.com/event/create?template=general_dyo&event_type=&c=featured&ctrk=featured&op_exp=quick_create&expand_gallery>

* 1. **Overview**

This project is an online event management where users can invite guests to attend their event. For example, if anyone want to send any invitation for an event it requires a lot of effort and time and there are very few applications to do this. Let’s assume that user Bob wants to host an event and want to invite his friend Lacy to that event. So, he can create an event of which type and other details in it and send it to Lacy. When Lacy opens that invitation, she can choose the RSVP whether she wants to attend it or not. This is the overview of this application.

**Unit 2: GENERAL DESCRIPTION**

* 1. **Product Perspective:**

This application is making the user to create or select any template or invitation and then after selecting the invitation, the user is going to send the invitation to all the guests through mail or text to the phone. The user can choose among the different invitations and send them using this application.

* 1. **Product Functions**

This application does not have any admins. It consists of only users. Users can create an invitation and invite the guests.

This application does not have any admins. It consists of only users. Users can create an invitation and invite the guests.

**User can perform all the below operations:**

* Sign up and log in to the application.
* If the user forgets the password, then the user can reset the password.
* Can choose the type of invitation.
* Ability to create the groups so that next time they can be invited very easily.
* Responses from the guests can be taken as the preferred food, no.of guests coming, etc.
* Users can create a poll and take the responses from the guests.

**Guests can perform the below Operations:**

* Able to login to the application.
* Can view all the invitations they’ve been invited to.
* Answer to the polls for which the hosts have created.
* Manage all the invitations.
* Giving the RSVP

**2.3 User Characteristics**

* This application can be used by any users to create any invitation and invite to any event.
* User can give the response like attending, not or may be later which get recorded.
* User(host) can also create a poll like what to bring and also any additional information if required.

* 1. **General Constraints**
* While using the servers there might be a chance of application not working properly.
* Only guests can create an event.
* Hosts should give the confirmation like attending the event or not.
  1. **Assumptions and Dependencies**

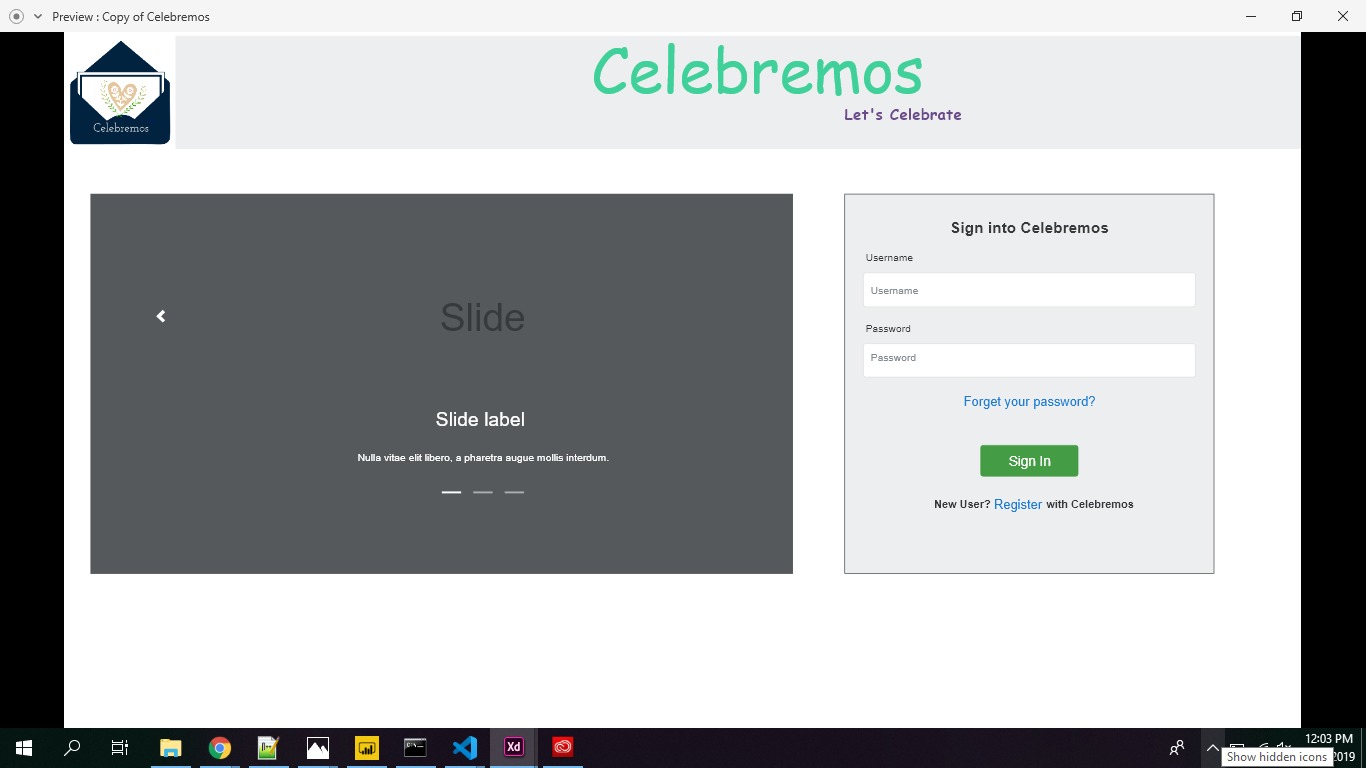
The people who are working in this project have a clear idea of the requirements.

**Unit 3. SPECIFIC REQUIREMENTS**

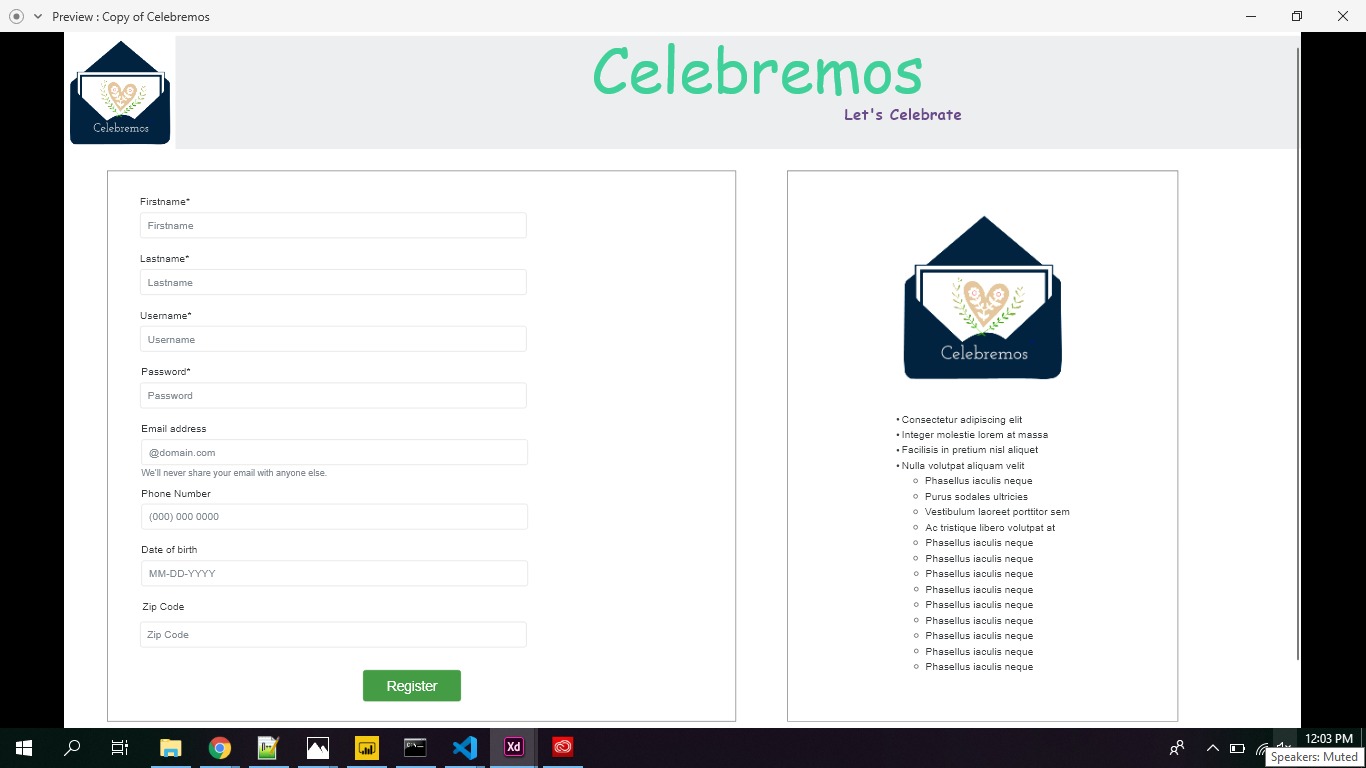
* 1. **External Interface Requirements** 
     1. **User Interfaces**

User interfaces used in this project are mentioned below, welcome screen, sign-up screen, sign-in screen of user, sign-in screen of admin.

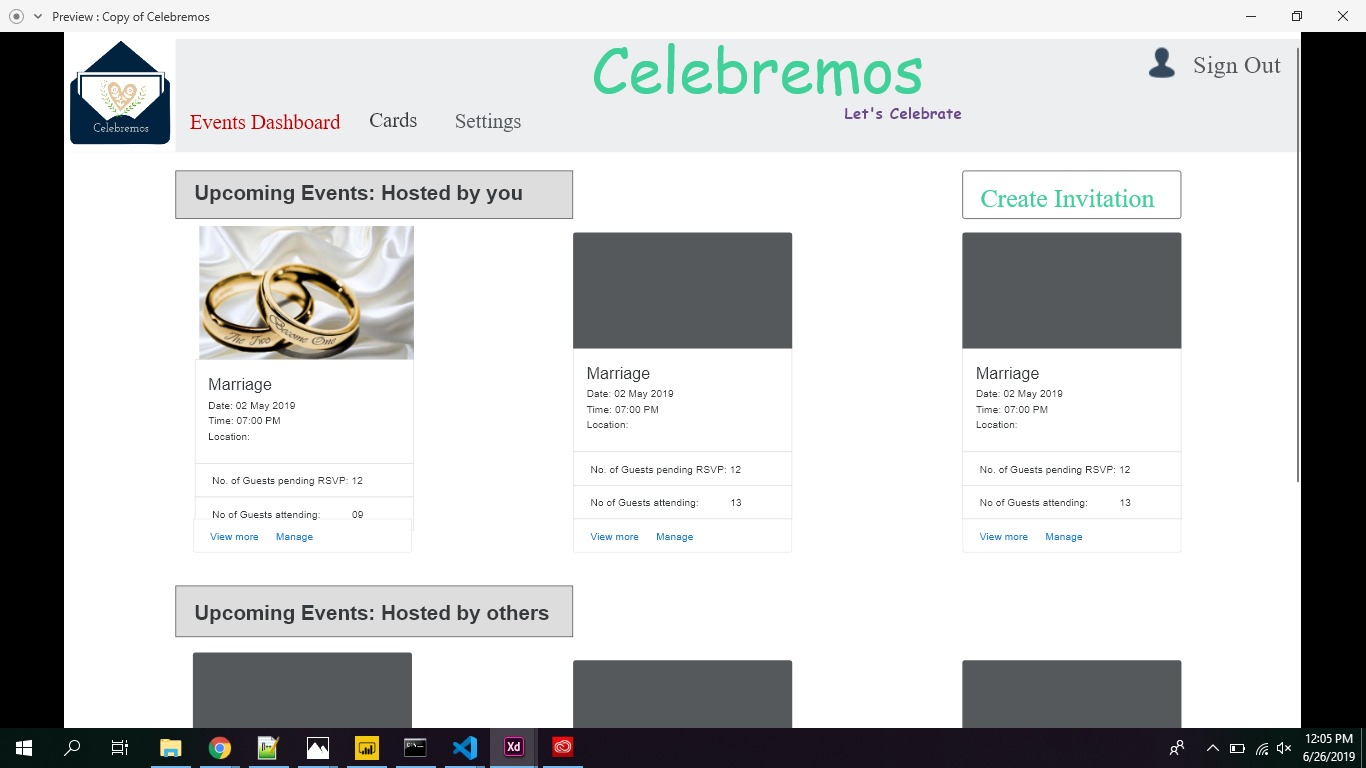
1.User login screen

****

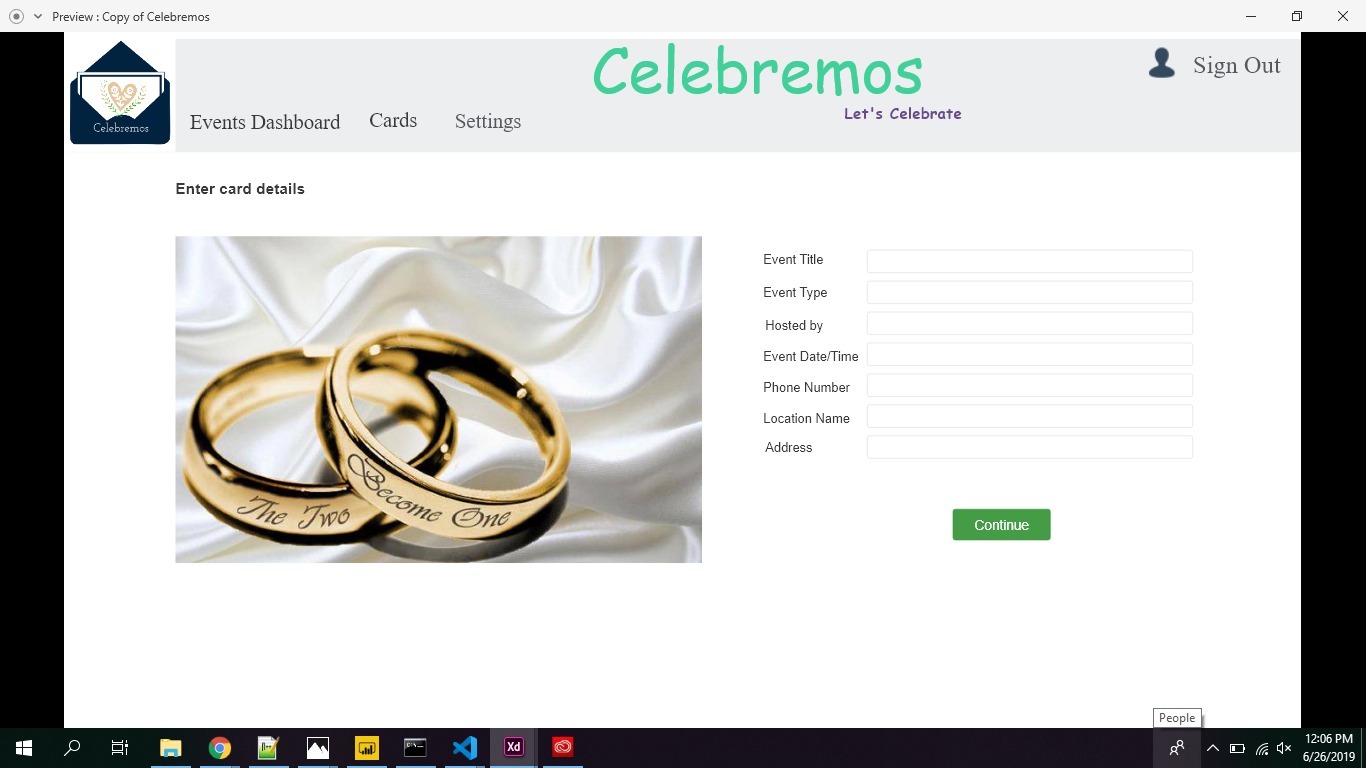
2.Sign-up screen

****

3.Event Dashboard screen

****

4.Event Creation screen

****

* + 1. **Hardware Interfaces**

Our hardware interfaces are mobile phones with browser in it. Memory should be of minimum 32 GB. Operating system can be of both windows and MacOS. The minimum processor can be 2.3 GHz dual-core Intel Core i5.

* + 1. **Software Interfaces**

Visual Studio Code is used to develop the entire application by writing code in it.

**Visual Studio Code** is a source-**code** editor developed by Microsoft for Windows, Linux and macOS. It includes support for debugging, embedded Git control and GitHub, syntax highlighting, intelligent **code** completion, snippets, and **code** refactoring.

**3.1.4 Communications Interface**

**Google Docs, sheets and slides:**

Google Docs, sheets and slides lends itself to collaborative projects in which multiple authors work together in real time from geographically diverse locations. All participants can see who made specific document changes and when those alterations were done. Because documents are stored online and can also be stored on users' computers, there is no risk of total data loss as a result of a localized catastrophe. However, the Internet-based nature of Google Docs, sheets and slides has given rise to concerns among some authors that their work may not be private or secure.

We use this for creating project documents, sheets and slides for presentation. It is reliable for teamwork.

**Outlook:**

It is Microsoft e-mail service. We use outlook to send and receive mails. Used to contact team members.

**3.2 Functional Requirements**

**3.2.1 Host Modules**

1. Register
2. Login
3. Send an invitation
4. Modify/Change an invitation
5. Cancel an invite
6. View card templates
7. Save the invitation for later use
8. View the number of hosts who confirm
9. Add a poll to an invitation
10. What to bring?
11. Add food menu
12. Add guests to the contact list
13. SMS invitation
    * 1. **Invitee Modules**
14. RSVP invite
15. Register
16. Login
17. View the invite
18. Similar functionalities as host

**3.3 Use Cases**

A use case is a methodology used in system analysis to identify, clarify, and organize system requirements. The use case is made up of a set of possible sequences of interactions between systems and users in a particular environment and related to a particular goal. It consists of a group of elements (for example, classes and interfaces) that can be used together in a way that will have an effect larger than the sum of the separate elements combined. The use case should contain all system activities that have significance to the users. A use case can be thought of as a collection of possible scenarios related to a particular goal, indeed, the use case and goal are sometimes considered to be synonymous.

A use case (or set of use cases) has these characteristics:

* Organizes functional requirements
* Models the goals of system/actor (user) interactions
* Records paths (called *scenarios*) from trigger events to goals
* Describes one main flow of events (also called a basic course of action), and possibly other ones, called *exceptional* flows of events (also called alternate courses of action)
* Is multi-level, so that one use case can use the functionality of another one.

In our project, our use cases are user details, guests, invitation cards, list of events etc., You can see all use cases in Use case diagram section.

**3.4 Class/Objects**

* User:

To store the user details like name, email, password, type of user.

* Create Invitation:

To create the invitation and send it to the invitees.

* Save Invitation:

To save the invitation for later use.

* SMS Invitation:

SMS invitation is used to send invitation through messages.

* Import Contacts:

Import contacts is used to import contacts from a csv file.

**3.5 Non-Functional Requirements**

**3.5.1. Performance**

This is to check the speed of the app that means how fast the app is responding to the actions. Generally, every action should not take more than 15 seconds to respond. In this application the user should not feel any delay in response when he performed the series of action through the application.

**3.5.2. Reliability**

Reliability is the percentage of time that the app works correctly without any failures and to deliver the results properly. For our app it is the amount of time the app can run process of selecting products and make transaction of it. It should be 99 percentage in general.

**3.5.3. Availability**

Availability means the provider systems should meet the agreed availability targets like service downtime and available hours as defined in the operational level agreement. For this app we will specify the downtime periods for month and the services will be not available at that time and rest of the time the services are available.

**3.5.4. Security**

The provider should resist unauthorized access through the application and should provide access to authorized users only. So, in this app we will authenticate the users based on security mechanisms and provide security for the user’s personal information.

* + 1. **Portability**

Portability checks for whether the app works for different platforms or different OS environments.

* 1. **Inverse Requirements**

Inverse requirements state the requirements what the system will not do. It describes the constraints on allowable behavior. In most of the cases, it is easier to state that certain behavior must never occur than to state requirements guaranteeing acceptable behavior in all circumstances. It depends mostly on the client.

* 1. **Design Constraints**

Most web products are failed because of not limiting the application design. Here are some design constraints that we are planning to use.

**1.Client-side storage**: From the statistics the average app loses more than 50% of the users within 3 days because of the storage problem that users are facing.

2.**Update in the Application**: If the update has any problems then there’s a chance of reduction in the users. So, the update must be fully tested and should be bug free.

* 1. **Logical Database Requirements**

Specify the logical requirements for any information that is to be placed into a database, like

* Types of information used by various functions
* Frequency of use
* Integrity constraints
* Data entities and their relationships
  1. **Other Requirements**

A Mac Book with minimum requirements like 8 GB RAM and 500 GB hard disk and should be able to run X-Code. An iPhone with minimum model of iPhone 6 is required for testing the application. IOS developer account is required to access more granular control over the operating system entities. A NoSQL database for storing data related to the project.

* 1. **Prototypes (for complete project)**

Prototype is an early sample, model, or arrival of an item worked to test an idea or process or to go about as a thing to be repeated or learned from.

**Categories of Prototypes:**

**Proof-of-Principle Prototype** serves to verify some key functional aspects of the intended design, but usually does not have all the functionality of the final product.

**A Working Prototype** represents all or nearly all the functionality of the final product.

**A Visual Prototype** represents the size and appearance, but not the functionality, of the intended design. A Form Study Prototype is a preliminary type of visual prototype in which the geometric features of a design are emphasized, with less concern for color, texture, or other aspects of the final appearance.

**A User Experience Prototype** represents enough of the appearance and function of the product that it can be used for user research.

**A Functional Prototype** captures both function and appearance of the intended design, though it may be created with different techniques and even different scale from final design.

**A Paper Prototype** is a printed or hand-drawn representation of the user interface of a software product. Such prototypes are commonly used for early testing of a software design, and can be part of a software walkthrough to confirm design decisions before more costly levels of design effort are expended

**3.11. Use Case Diagrams**

A UML use case diagram is the primary form of system/software requirements for a new software program underdeveloped. Use cases specify the expected behavior(what), and not the exact method of making it happen (how). Use cases once specified can be denoted both textual and visual representation (such as UML). A key concept of use case modeling is that it helps us design a system from end user's perspective. It is an effective technique for communicating system behavior in the user's terms by specifying all externally visible system behavior.

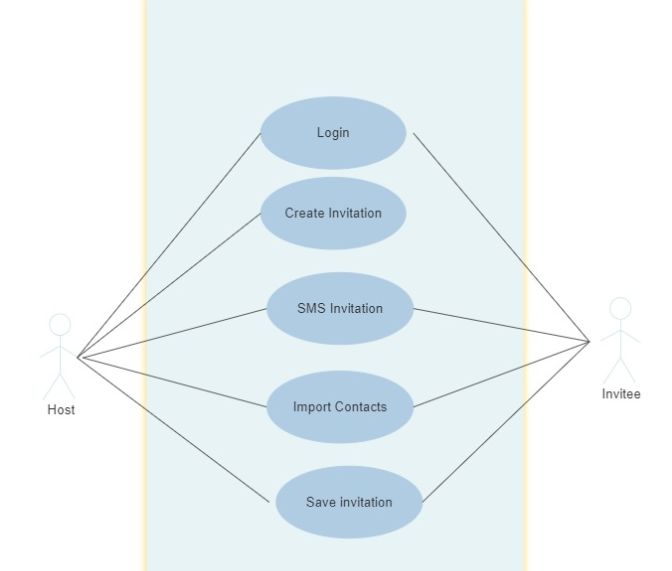


Fig 3. Use-Case Diagram

**UNIT 4. DESIGN**

**4.1. ER diagram:**

An entity-relationship model (ERM) is a theoretical and conceptual way of showing data relationships in software development. ERM is a database modeling technique that generates an abstract diagram or visual representation of a system’s data that can be helpful in designing a relational database. These diagrams are known as entity-relationship diagrams, ER diagrams or ERDs.

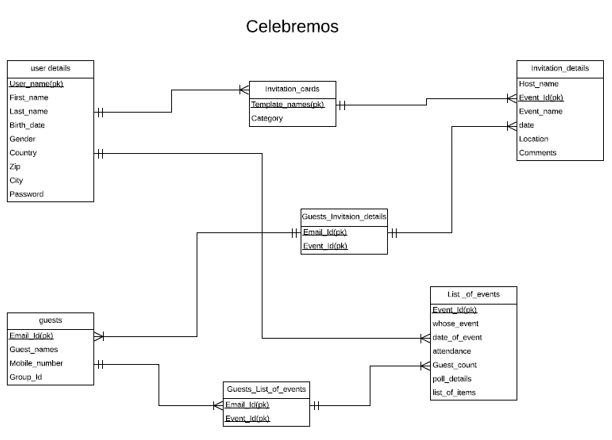
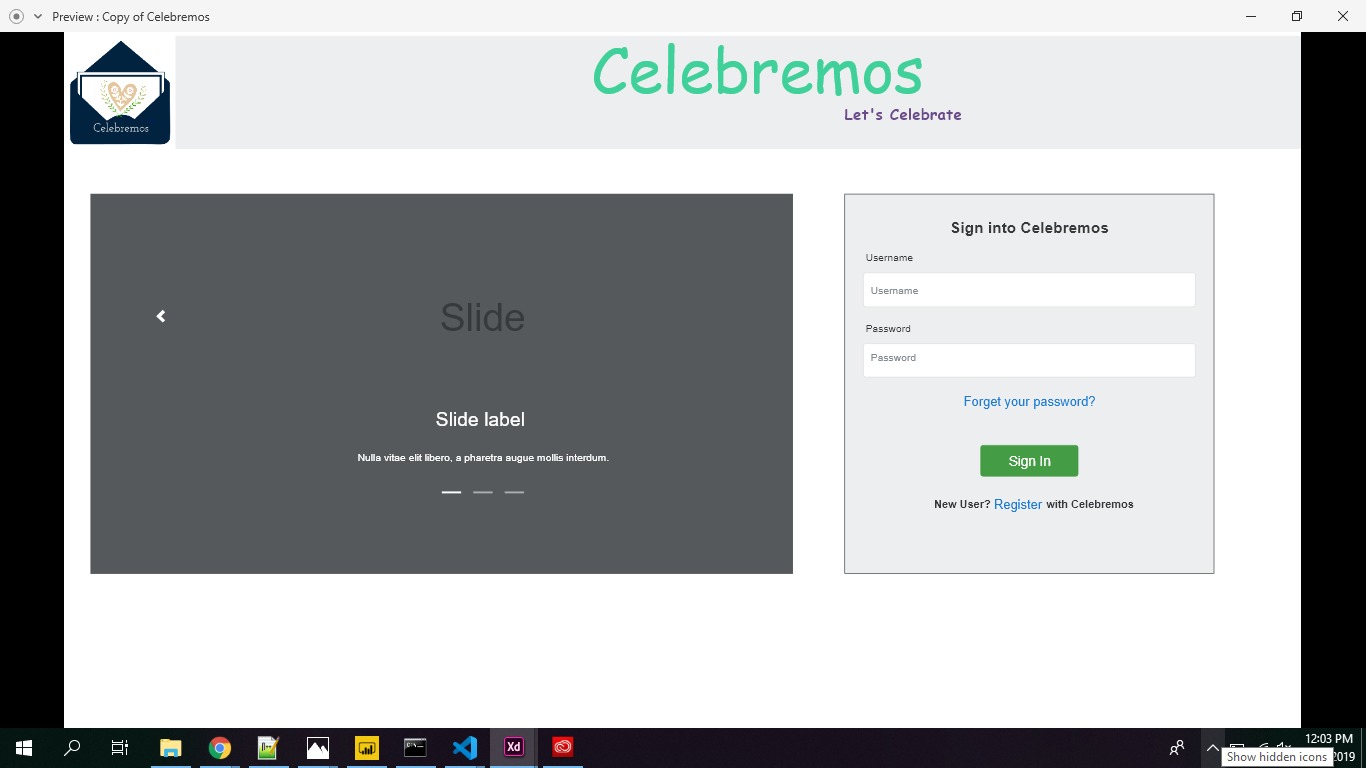


Fig 4. ER Diagrams

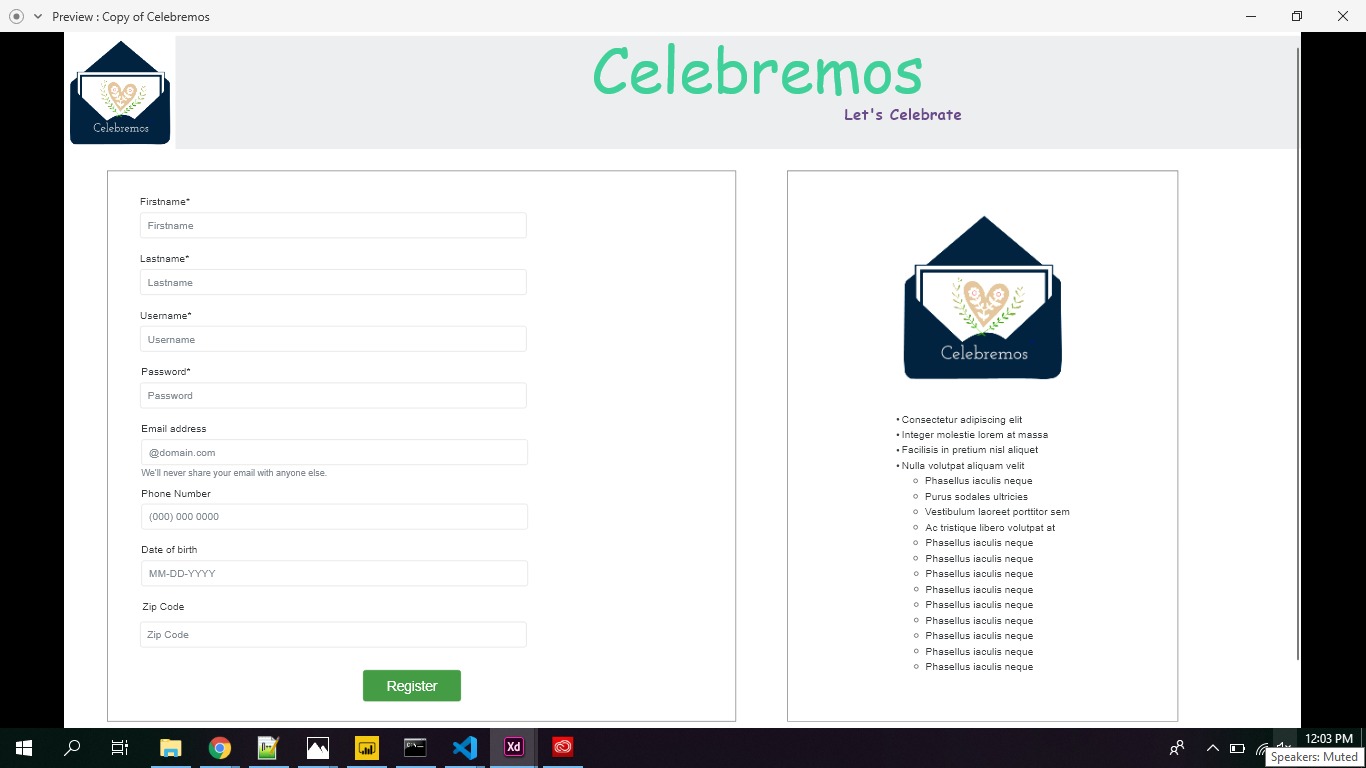
**4.2. GUI:**

Adobe XD reflect the design choices for color schemes, layouts, typography, iconography, the visuals of navigation, and the overall atmosphere of the product.

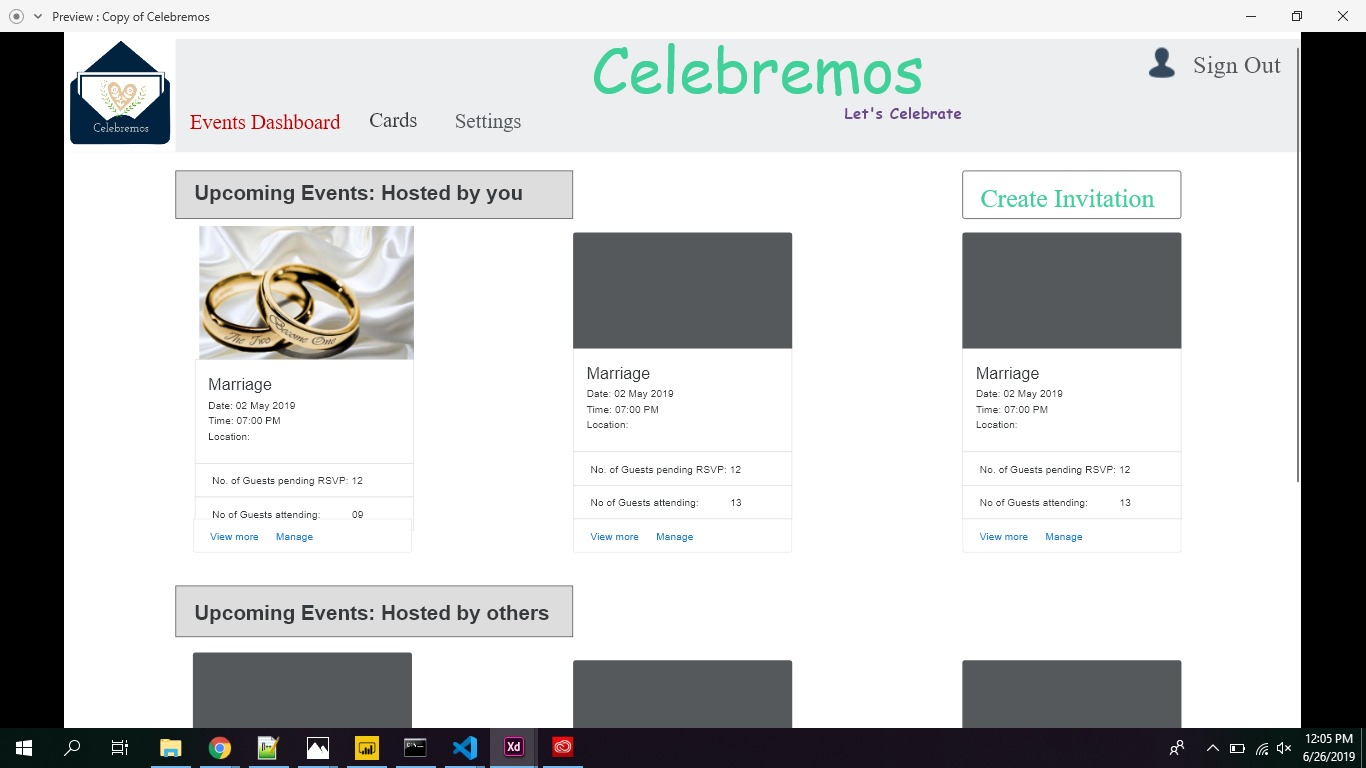
User login screen

****

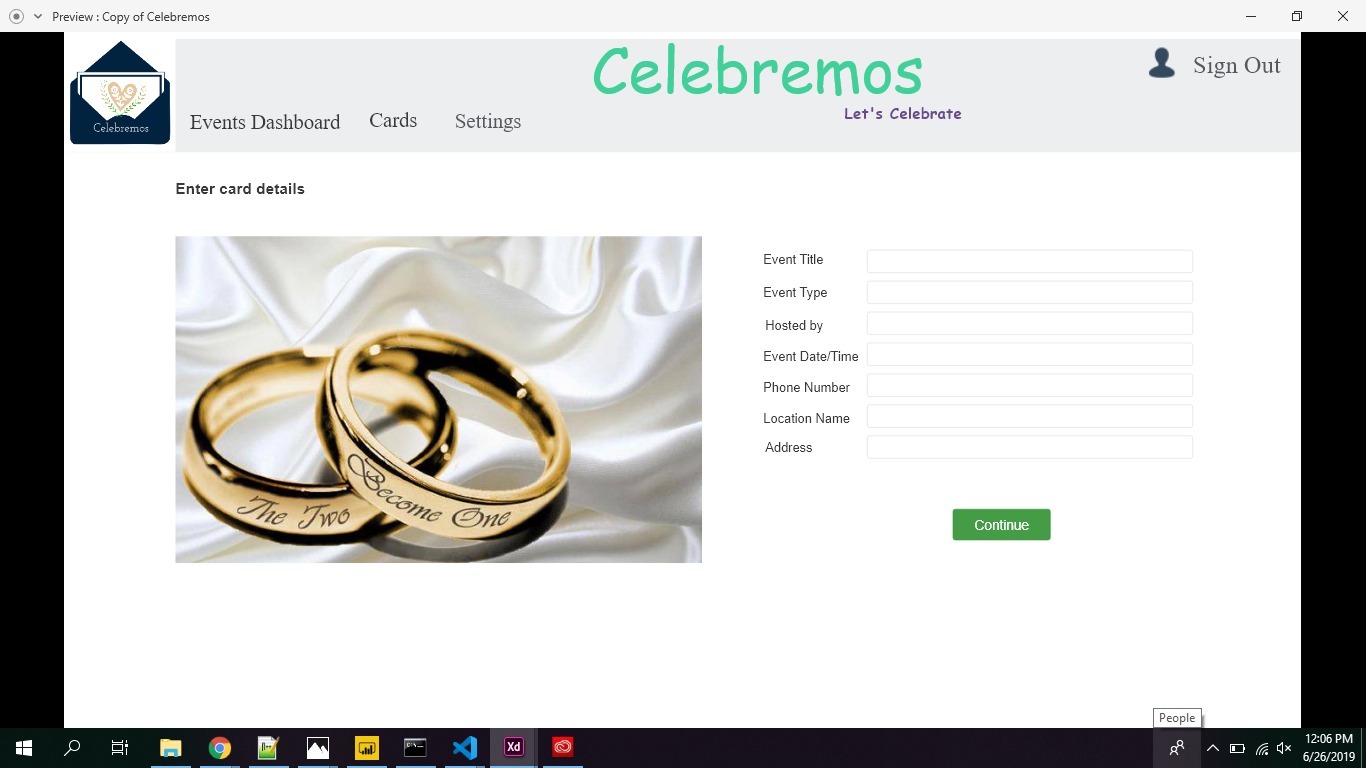
Sign-up screen

****

Event Dashboard screen

****

Event Creation screen

****

**Unit. 5: ANALYSIS MODELS**

**5.1. Data Flow Diagram**

A **data-flow diagram** (DFD) is a way of representing a flow of a data of a [process](https://en.wikipedia.org/wiki/Process) or a system The DFD also provides information about the outputs and inputs of each entity and the process itself. A data-flow diagram has no control flow, there are no decision rules and no loops. Specific operations based on the data can be represented by a [flowchart](https://en.wikipedia.org/wiki/Flowchart).

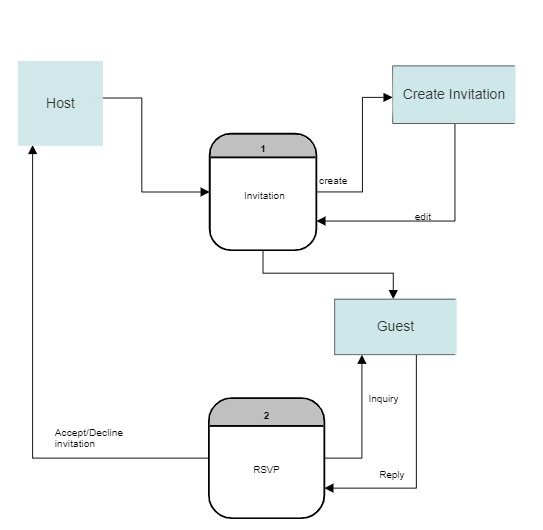


Fig. 9: Dataflow diagram

**5.2. Sequence Diagram**

A sequence diagram shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario. Sequence diagrams are typically associated with use case realizations in the Logical View of the system under development. Sequence diagrams are sometimes called **event diagrams** or **event scenarios**.

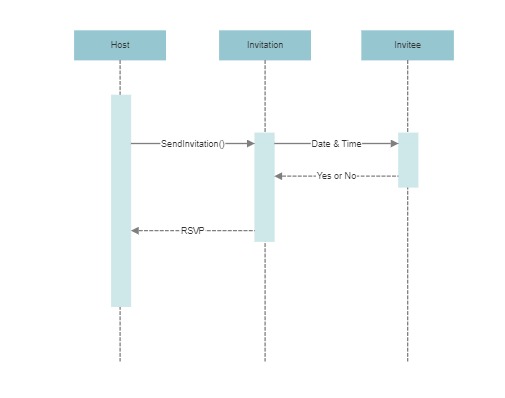
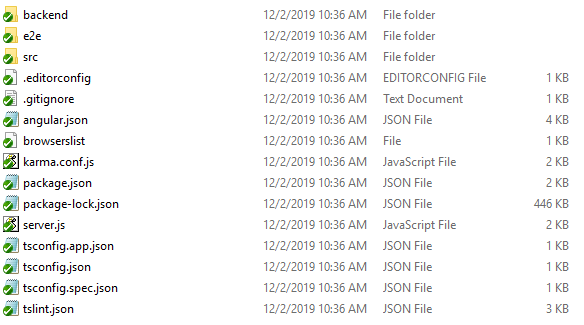


Fig. 10: Sequence Diagram

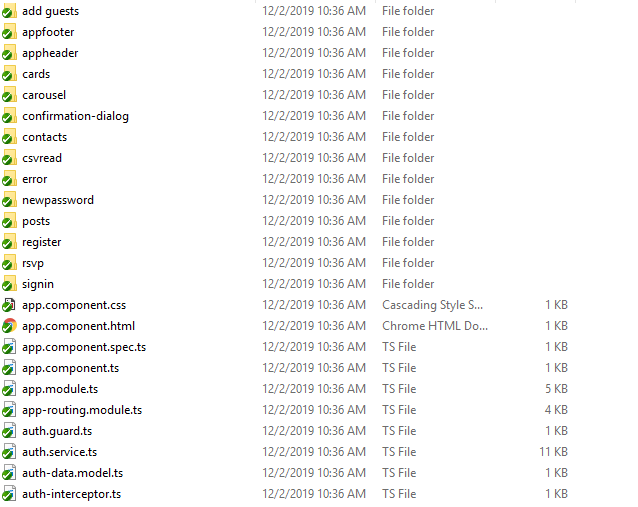
**UNIT 6. Technical Manual**

Technical Manual contains the detailed instructions on how to build an application from scratch.

Here is the folder structure for the application.



Here is the screenshot of all the components that we have used to develop the application.



**6.1 Login page**

This page has username and password text fields. When user enters username and password verifies user credentials.

* The Login authorization of user is done. We have UI validations for username and password and will displays proper feedback to user.
* Forgot password functionality is implemented and a link is sent to the user through the e-mail address provided and password can be reset using that link.



**6.2 Registration**

User must register with required details. When user clicks on Registration Button, if all the details given are correct then feedback message user is sent “Registration Successful”. If the details are wrong, an error message is displayed to the user on the screen.

All the Valid details are stored in the database when the user is successfully registered.



**Host Functionality**

**6.3 Create Events**

This create post is used to create an event with couple of details that can be filled in a form. Couple of validations are performed while filling this form out and the data must be given to the mandatory fields.

**6.3.1 Add Guests**

This module is still what we are working on where the main functionality of this module is to add guests and send them the invitation at a time.



**6.3.2 Save Invitation for Later Use**

After creating an event we can save the invitation for later use.



**6.4 Dashboard**

All events can be managed here.

**6.4.1 All Events**

Here all events that are hosted, invited and saved are present as reference.



**6.4.2 Hosted Events**

Here all events that are hosted by the host are present.



**6.4.3 Saved Events**

Here all events that are saved for future use are present.



**6.4.4 Invited Events**

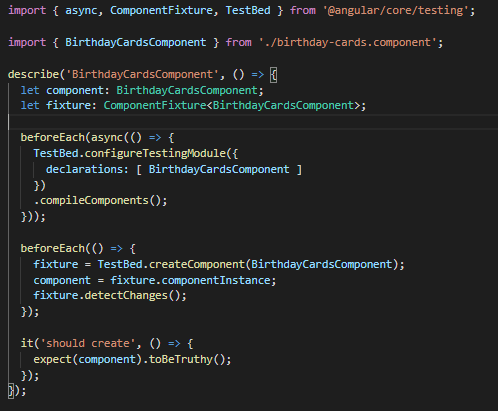
Here all the events that are you are invited are present.



**6.5 Cards and Contacts**

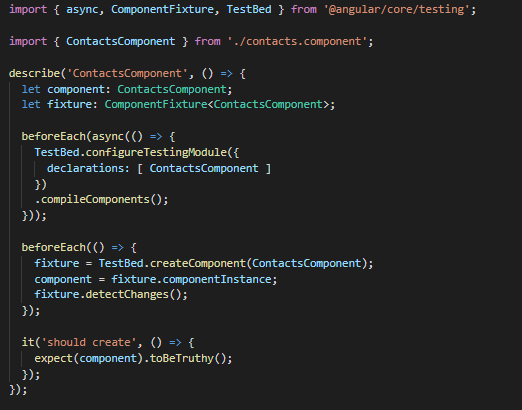
**6.5.1 Cards**

Here we can find various pre-defined templates for different events such as birthdays, weddings etc.



**6.5.2 Contacts**

Contacts with first name and last name can be added to a group which can be used later while sending an invitation.



**Invitee Functionality**

**6.6 RSVP**

RSVP is like taking count of the number of guests that are attending the event and making the arrangements according to the comments given in the registration form.



**6.7 Database Functionality**

Database used for this application is “MongoDB”.

MongoDB stores data in flexible, JSON-like documents, meaning fields can vary from document to document and data structure can be changed over time

Using MongoDB, we have implemented login, registration and forgot password functionalities.

**6.8 Messaging:**

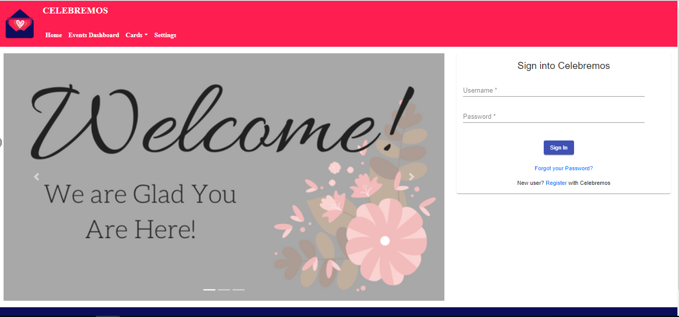
SMS invitation can be sent to the users, which is very flexible to view when the users are busy in other works.

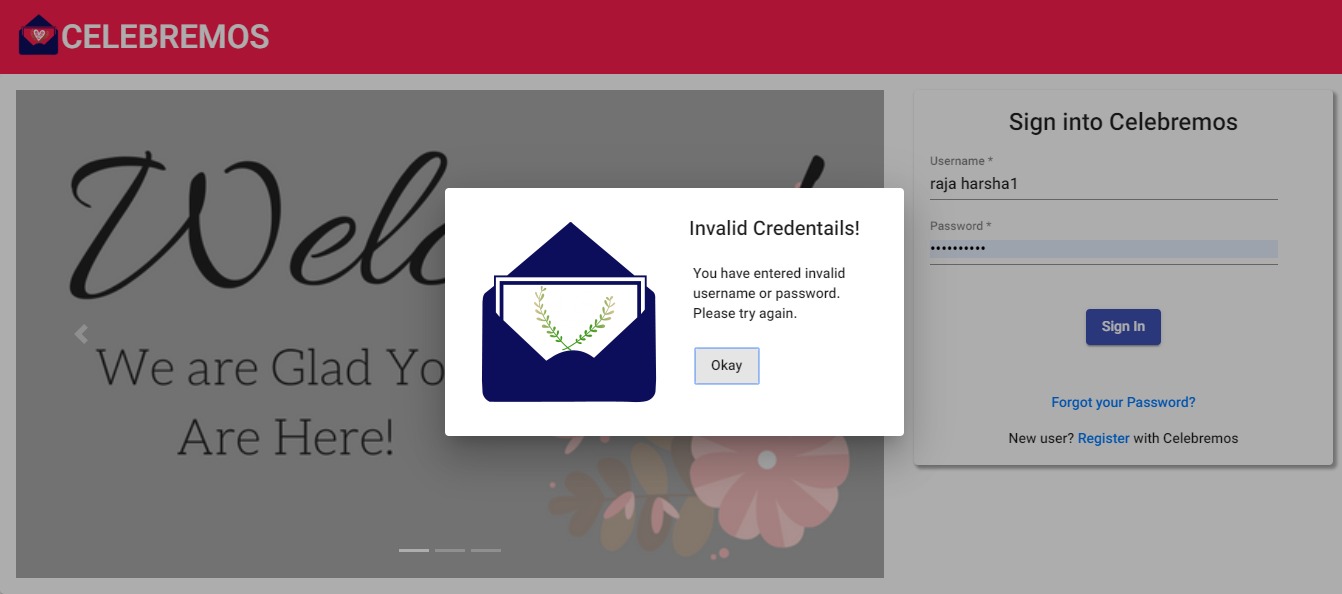
**UNIT - 7 END USER MANUAL**

**7.1 Login page:**

Every user need to login to access all the functionalities. If he/she is a new user, they need to sign-up for new account. Every user is recommended to have an account. If user forgets his password, can reset his password using “Forgot password” functionality.

Login screen consists of two fields, one is username and other is password. user needs to give his correct username and password in respective fields. If the given credentials are valid, user can log-in. If credentials are invalid, they are acknowledged with a feedback message “Invalid username and password”.



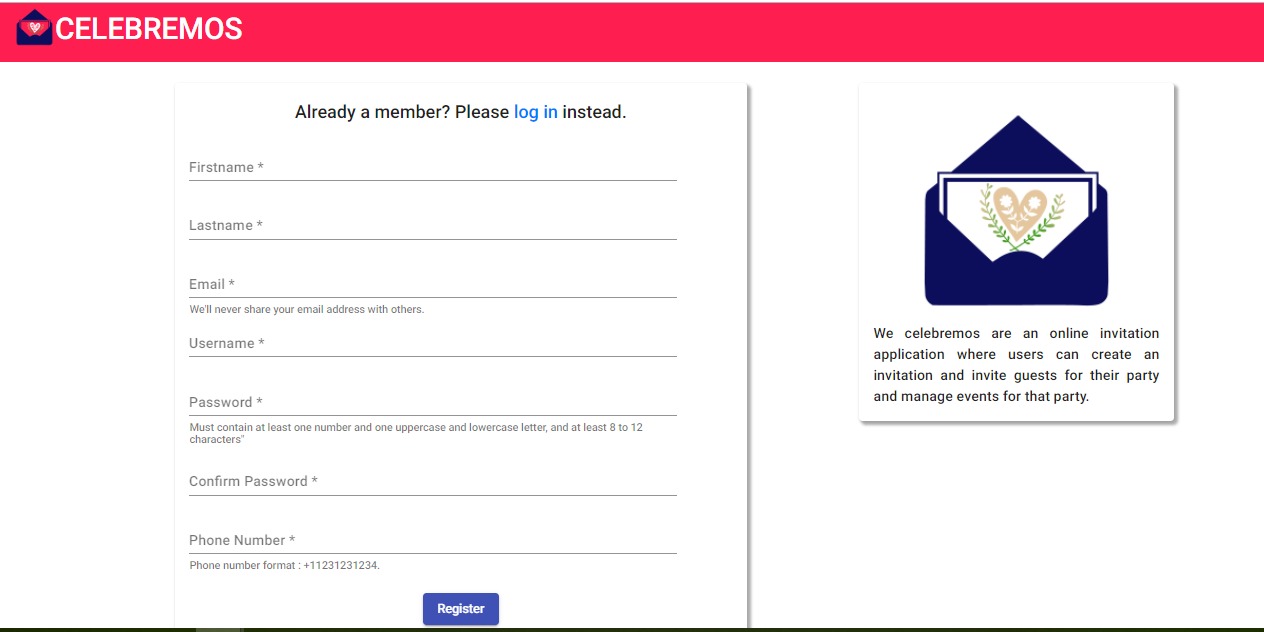


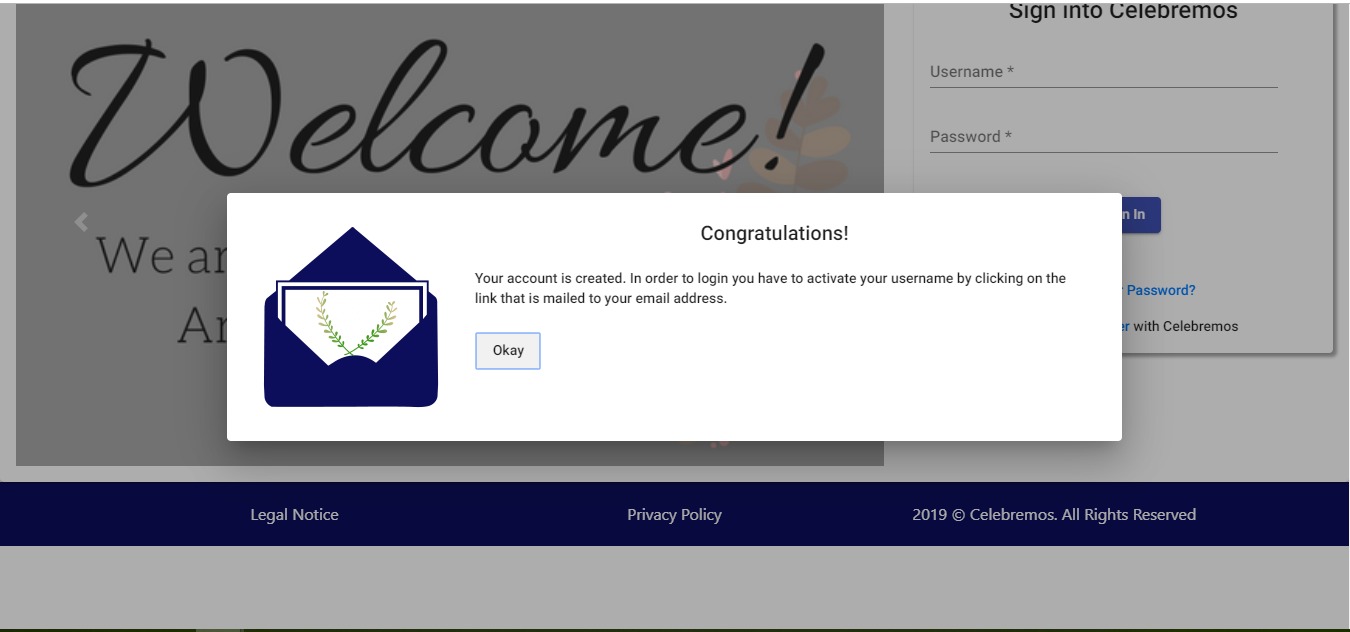
**7.2 Registration:**

When user clicks on “Sign-up” in log-in screen, user is redirected to registration page. In registration page user needs to fill all required fields for successful registration.

User needs to fill Name, e-mail address, phone number, password to get registered. User E-mail address is used as username for the account. Password should have at least one capital letter, one special character, one number, and minimum eight characters in total.

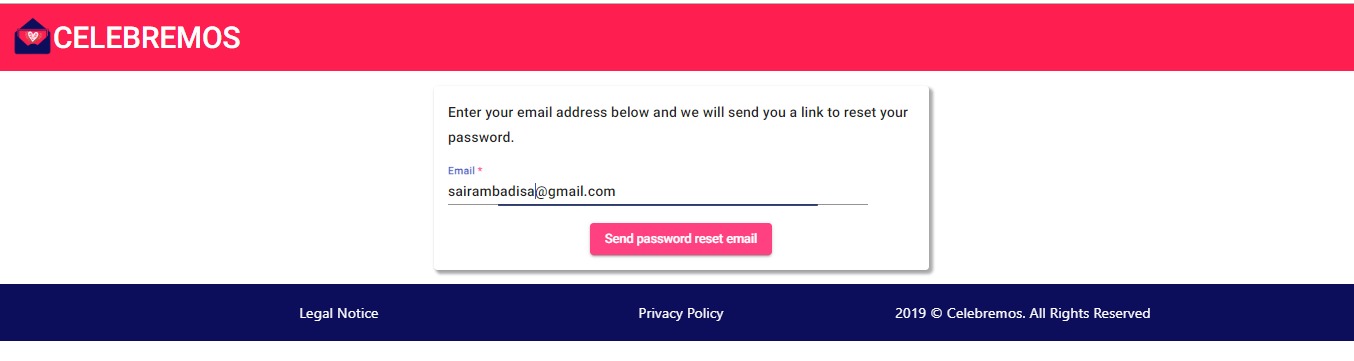
If the required fields are filled and valid, user is registered successfully. Otherwise, user receives a feedback message that he/she is not registered successfully.

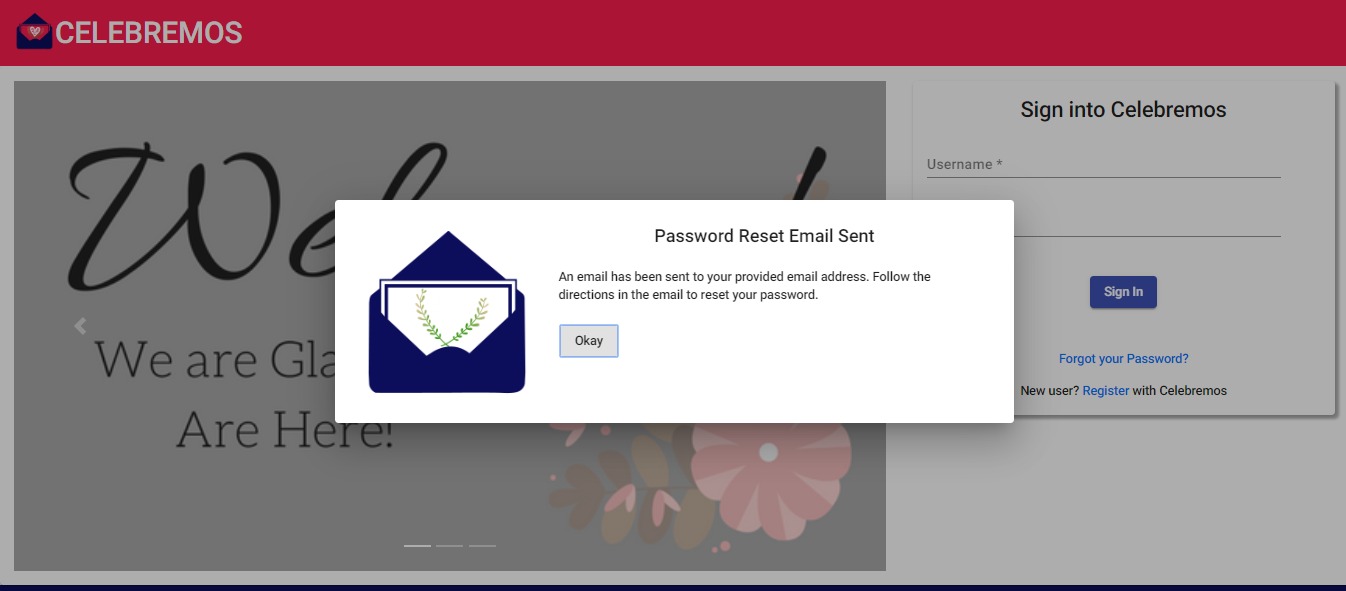




**7.3 Forgot Password:**

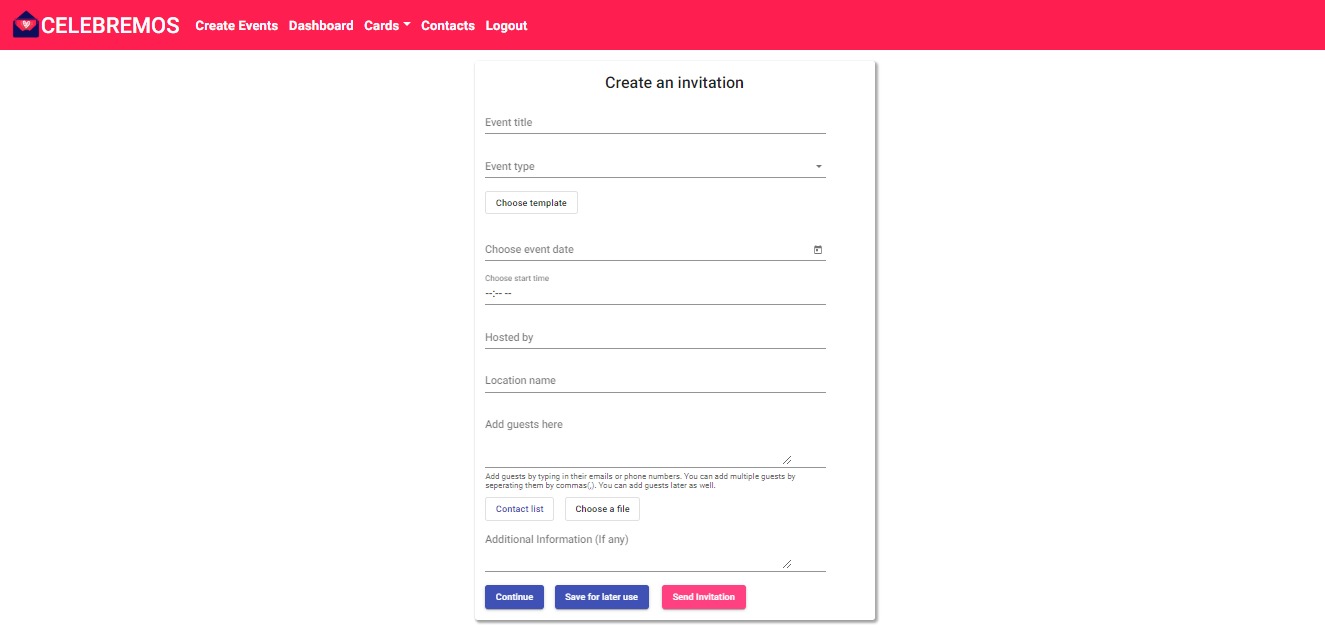
When user clicks on forgot password, he/she is redirected to forgot password screen. In forgot password user needs to give his registered e-mail address in requested field. If the e-mail is registered e-mail, user was sent a link to reset password. By clicking on that e-mail, user can reset his password.





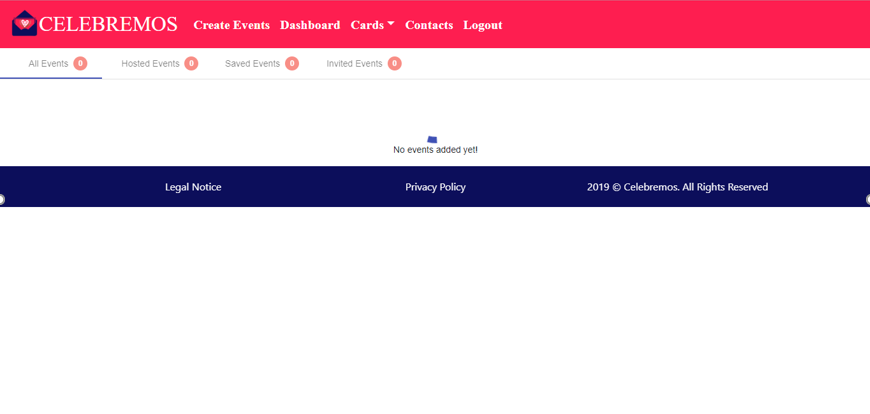
**7.4 Create Events**

This create post is used to create an event with couple of details that can be filled in a form. Couple of validations are performed while filling this form out and the data must be given to the mandatory fields.



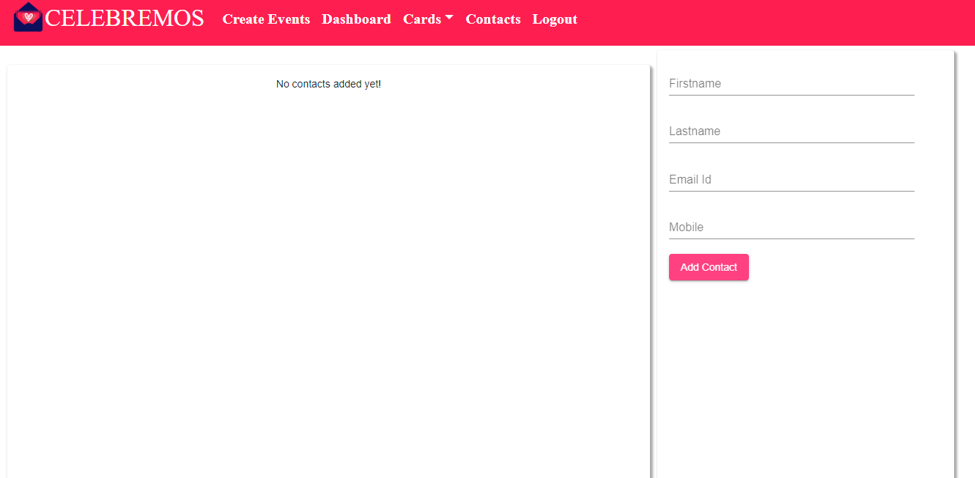
**7.5 Dashboard**

Dashboard components is one of the important components which displays the dashboard with the past events, invited events and events which are saved for the later use.



**7.6 Cards and Contacts**

In order to send the requests, we should add their contact numbers or the mail id’s of the individual person where we can directly send the invitation through mail while creating an event or by adding the guests.



**7.7 RSVP**

RSVP is like taking count of the number of guests that are attending the event and making the arrangements according to the comments given in the registration form.