****

SOFTWARE DESIGN AND ANALYSIS PROJECT REPORT

EVENT HUB

**GROUP MEMBERS**

1. 18K-0158 Muhammad Sameer
2. 18K-0272 Shahzaib Bhutto
3. 18K-1163 Muneeb Asif
4. 18K-0173 Muhammad Hassan Ali Khan

INDEX

[1.1 INTRODUCTION](#_7cnoo9mhylr)

[1.2 MAIN OBJECTIVES](#_30j0zll)

[1.3 PROJECT OVERVIEW](#_1fob9te)

[1.4 SYSTEM ANALYSIS](#_3znysh7)

[PROBLEM STATEMENT](#_2et92p0)

[1.5 STAKEHOLDERS](#_z8unca8gf1m1)

[1.6 MAIN FUNCTIONS](#_3dy6vkm)

[REGISTER NEW USERS](#_1t3h5sf)

[AUTHENTICATION](#_4d34og8)

[SEARCH AND UPDATE EVENTS](#_2s8eyo1)

[PAYMENT AND RESERVATION](#_17dp8vu)

[1.7 SOFTWARE REQUIREMENTS](#_i66xg3klnbha)

[SYSTEM REQUIREMENTS (NON FUNCTIONAL)](#_odf598kwgpb3)

[USER REQUIREMENTS (FUNCTIONAL)](#_8om94vdo4c0m)

[1.8 DIAGRAMS](#_40aa3k3c7rzm)

[CLASS DIAGRAM](#_a9urpfhmo27p)

[USECASE DIAGRAM](#_ef79q6mz1a96)

[USE CASE NARRATION](#_b52z2zhb4zg5)

[ECB ROBUSTNESS DIAGRAM](#_4k3jhgwf87a3)

[ACTIVITY DIAGRAMS](#_pkq76km7kbcg)

[SEQUENCE DIAGRAM](#_uxqyx1172lwj)

[ADD EVENT](#_tehf1nzbv24b)

[LOGIN](#_txuhmpwrgw8l)

[COLLABORATION DIAGRAM](#_i81lz3xul60x)

[ADD EVENT](#_go0xbxtnnzl5)

[LOGIN](#_cm68tzrgaoj3)

[STATE TRANSITION DIAGRAM](#_31rrgha5l6ef)

[STATE DIAGRAM #1](#_q1zwksd7taj6)

[STATE DIAGRAM # 2](#_prp2xi982v8b)

[STATE DIAGRAM # 3](#_w2eyhq81fbbl)

[DEPLOYMENT DIAGRAM](#_fkk69si8n4pa)

# 

# **1.1 INTRODUCTION**

Event management system is used to manage all the activities included in organizing an event, activities include everything from booking an event to hosting an event on a particular date selected by the user. In any event there are multiple important jobs needed to be taken care of simultaneously and in order to help the user in managing these jobs we have developed this software. With the help of this software a customer can easily contact the event management team or other event planners who work privately as a freelancer. This system helps the event management company by managing their paper works and by generating an online report for the jobs they have completed and saving the details of their successful events in a separate database dedicated for the system.

Overall this project of ours is developed to help the customers, staff of the event management team and all the other private event planners in organizing their events and to reduce the overall human efforts.

# **1.2 MAIN OBJECTIVES**

* To develop a system that maintains the data storage and retrieval in an easier way.
* To provide a platform for the underdeveloped private event planners in running their business.
* To record the services provided by the team and all the events organized by the team
* To develop a system that requires minimal user training and interaction with the system is easy.
* To store the records of the users registered in the system and to restrict the access of modifying and deleting the record to the admin.

# **1.3 PROJECT OVERVIEW**

This project is an example of a project management system, this project will help people in finding and booking halls for their events at a given time slot. A customer can also get all the event details that are currently active on the site, he can also book tickets of the current events that are scheduled. Whenever a user logins or registers on the site he can add an event, by hiring a professional. The event needs to be approved by the admin before getting registered on the site. After the event is registered the customer is asked to pay the advance fee of the event before the event is listed on the site. After the payment is processed the event gets listed on the site if it is a public event. All the records are stored in the database in the backend.

# **1.4 SYSTEM ANALYSIS**

## **PROBLEM STATEMENT**

The main problems that occurred before having a computerized system included:

* If a computerized system is not implemented a data file can get easily lost, sometimes due to a human error or loss of data in case of any natural disaster
* Keeping a backup copy of the data in case of any natural disaster would be very difficult if a computerized system is not implemented, as copying the data would be very difficult if the number of data entries is large and the data is not digitally stored.
* Searching for data would be also very difficult if the data entries are not properly sorted. The searching of data would be impossible in case of large unsorted data.
* The professionals who wanted to work for the team would face difficulty in contacting and collaborating with the team.
* The overall coordination between the user and the team would be difficult and monitoring the transactions would also be a big problem if no proper computerized system is implemented.

# 

# **1.5 STAKEHOLDERS**

These are the users which are directly associated with the system. These users are interacting with the system and are responsible for making changes in the system data entries.

1. **EVENT ORGANIZER:** This is a primary stakeholder who is responsible for planning the events and ensuring that these events run smoothly. They are responsible for handling the queries related to the events which are presented by the customers who want to schedule an event. They produce detailed proposals for the events and provide an estimated budget to the admin and charge the customer accordingly. They are responsible for researching the venues and negotiating a deal with the customer. After the registration as a normal user a person can apply for this job and become a part of the event management team.
2. **EVENT ADMINISTRATOR/ADMIN:** This is also a primary stakeholder who is responsible for managing the whole system as well as monitoring all the transactions which are happening in the system. The key features of this stakeholder also includes addition and deletion of new data entries regarding new events and users. The registration process and payment process both are verified by the admin user. Whenever an event is added by a user it is first approved by the admin.
3. **CUSTOMER/NORMAL USER:** A normal customer user who logs in the system and he can buy tickets for any currently listed events. He can also add an event which needs to be approved by the event administrator. A new user will have to register first in order to log in as a customer in the system.
4. **BROWSING USER:** A browsing user can only access a few panels of the system which includes the current events (show events) functions. In order to get full access to the system he needs to register in the system first.

# **1.6 MAIN FUNCTIONS**

## **REGISTER NEW USERS**

This feature is needed to be performed by all the users, in order to create an account and interact with the system. The EMS must be able to delete the information if the entered information is wrong.

## **AUTHENTICATION**

The system needs to verify the entered login credentials from the previously entered and saved user id and passwords. The system should match the entered details from the database and should allow access if they match.

## **SEARCH AND UPDATE EVENTS**

Users should be able to search the events and the admin can search and update the records as well as the events which are scheduled.

## **PAYMENT AND RESERVATION**

A system of payment and reservation of the tickets is also a part of the system. The normal user can select any of the ongoing events and he will be charged accordingly. Similarly, a customer who adds an event will be charged accordingly. A customer can edit his event plan and the number of tickets but the system will monitor all the transactions and update them accordingly.

# 

# 

# **1.7 SOFTWARE REQUIREMENTS**

These can be defined as the required conditions which are needed to be fulfilled, it deals with establishing the needs of stakeholders that are to be solved by the software. The software requirements can be further classified into two parts

1. System Requirements (NON FUNCTIONAL)
2. User Requirements (FUNCTIONAL)

## **SYSTEM REQUIREMENTS (NON FUNCTIONAL)**

The requirements which do not concern the overall functionality of the system are considered as system requirements.

* **RELIABILITY:** the implemented system should be reliable, I.E. it should not crash often and incase of system failure it needs to have the backup of the data.
* **PERFORMANCE:** the webpages needs to be loaded within a few seconds and incase of any error appropriate messages should be generated along with the solution of those problems.
* **SAFETY AND SECURITY:** The details need to be maintained properly and the users must be verified, the database should be kept backed up. The system needs to be highly secured with all the important functions privatized. The details of the user must be safe and secure and the details should only be accessible to the administrator.
* **IMPLEMENTATION REQUIREMENTS:** the project should consist of a front end with proper functions and it should be connected with the database. Any server side scripting language can be used as long as it is completing the job.
* **DELIVERY REQUIREMENT:** The project is expected to deliver in 4 months of time of the semester.

## 

## 

## 

## **USER REQUIREMENTS (FUNCTIONAL)**

* **USER LOGIN:** In order to enter in the system the user needs to login and verify their user id and password before continuing.
* **REGISTERING A NEW USER:** A customer can browse all the events but if the user wants to book a ticket for any of the events he needs to register in the system in order to continue, after registering the browsing user must wait for the admin to validate the credentials and accept the registration.
* **ABOUT US:** A web page which should be visible to all types of users, this webpage will show the details of the team.
* **DASHBOARD:** This should be determined at the time of login. Different functions will be present on the dashboard depending on the type of user logging in the system.
* **SHOW USERS:** this will show the lists of users registered in the system. This function will only be accessible to the admin user.
* **REGISTERING AS A PROFESSIONAL:** If a normal user wants to work for the team as a professional he can apply, the application details are then forwarded to the admin which decides whether the applying candidate is fit for the team.
* **HIRING A PROFESSIONAL:** If a normal user wants to add an event he can hire a professional in order to help him in managing the event.
* **ADD EVENT:** A normal user can add an event after adding an event he should wait for the approval of the event from the event administrator.
* **PAYMENT:** the system should be able to manage the payments and record all the transactions taking place within the system.
* **SEARCH:** System must be able to search the database efficiently and should be able to generate precise results based on the keywords entered.
* **MANAGE USERS:** An event administrator is an only user who has the right to manage, add, delete and update the users.
* **SHOW EVENTS:** A show event function should also be made part of the system. This function should be visible to all the users.

# **1.8 DIAGRAMS**

## **CLASS DIAGRAM**

## 

## 

## **USECASE DIAGRAM**

This use case diagram is a graphic depiction of the interactions among the elements of our Event management system. It represents the methodology used in system analysis to identify and organize the functional requirements of our Event Management system. The main actors of the event management system in this use case diagram are Admin, Normal User, Browsing User, and Bank. All the users perform different types of use cases.

**RELATIONSHIP BETWEEN ACTORS AND USE CASES**

* **ADMIN**

The use cases of admin are delete user, show profession add user, show user, show events, login, and approve event. The approved event use case depends on the verification of payment which is to be paid by the normal user. Whenever the admin logs in a different interface is presented.

* **NORMAL USER**

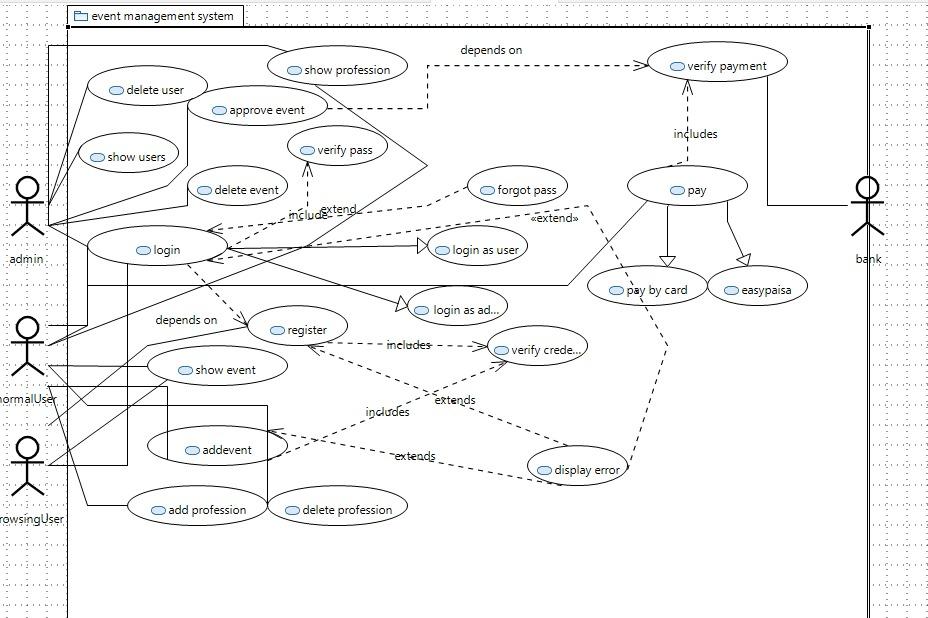
The use cases of a normal user are login, show events, show profession, pay, delete profession, and add event. The payment use case is further broken down into pay by card or locally and the payment must be verified by the admin in order to add an event.

* **BROWSING USER**

The use cases of browsing user includes show events, and login

* **BANK**

Bank is a secondary actor and the only use case of a bank is to verify payments which are transferred by a normal user.



## **USE CASE NARRATION**

|  |  |
| --- | --- |
| **Use Case Description:** To make a new account | |
| **Use Case Name:** Register | |
| **Primary actor*:*** System, User | **Other actors:** Database |
| **Stakeholders:** Company | |
| **Relationships**   * **Includes:** Verify credentials * **Extends:** Display error | |
| **Pre-conditions:**  None | |
| **Flow of Events:**  **Actors action**   1. User enters username. 2. User enters password and confirms password.   **5** User enters signup Button. | |
| **System response**  **2** System checks availability of username and proceeds.  **4** System checks if passwords match and proceeds.  **6** System creates new account. | |
| **Alternative and exceptional flows:**    **2.1** System displays username taken error if username already taken.  **4.1** System displays username taken error is passwords don’t match | |
| **Post-conditions:**  Account is created | |
| **Use Case Description:** To Login | |
| **Use Case Name:** Login | |
| **Primary actor*:*** System,User | **Other actors:** Database |
| **Stakeholders:** Company | |
| **Relationships**   * **Includes:** Verify Credentials * **Extends:** Display error | |
| **Pre-conditions:**  User must be registered | |
| **Flow of Events:**  **Actors Action**   1. User enters username 2. User enters password 3. User enters login | |
| **System Response**   1. The system validates the users password and logs him into the system 2. The system displays user dashboard and use case ends | |
| **Alternative and exceptional flows:**   * 1. Invalid username / password   *(If in the basic flow the system cannot find the name or the password is invalid, an error message is displayed. The actor can type in a new name or password or choose to cancel the operation, at which point the use case ends.)* | |
| **Post-conditions:**  None | |
| **Use Case Description:** To add events | |
| **Use Case Name:**  Add event | |
| **Primary actor*:*** User , Admin , System | **Other actors:** Database |
| **Stakeholders:** Company | |
| **Relationships**   * **Includes:** Verify Credentials * **Extends:** Display error | |
| **Pre-conditions:**  Must be logged in | |
| **Flow of Events:**  **Actors(User , Admin) action**    **1** User enter details.  **2** User press Add event button.  **4** Admin approves event. | |
| **System response**  **3** System adds event to pending table.  **5** System removes event from pending table and adds it to event table.  **6** System sends an acceptance notification. | |
| **Alternative and exceptional flows**  **3.1** System displays error if invalid details.  **4.1** Admin rejects event  **5.1** System removes event from pending table  **6.1** System sends rejection notification. | |
| **Post-conditions:**  Payment for the event required | |

# 

# 

## **ECB ROBUSTNESS DIAGRAM**

The project is based on three main user types and has separate dashboards for all the users. The starting interface of the system contains three main pages:

* **Active Events**

It will show all the active events at that time.

* **About Us**

It will show the details of the management team and will show the steps required if a user wants to become a part of their team.

* **Login**

The login page interface for all the users would be the same but after the user logs in only specific functions would be visible for the users according to the type of user that has logged in.After logging in the system will perform a verification of the credentials entered and if the entered credentials are not present in the database, the program will generate an error message. There are three types of users

* **Normal User**

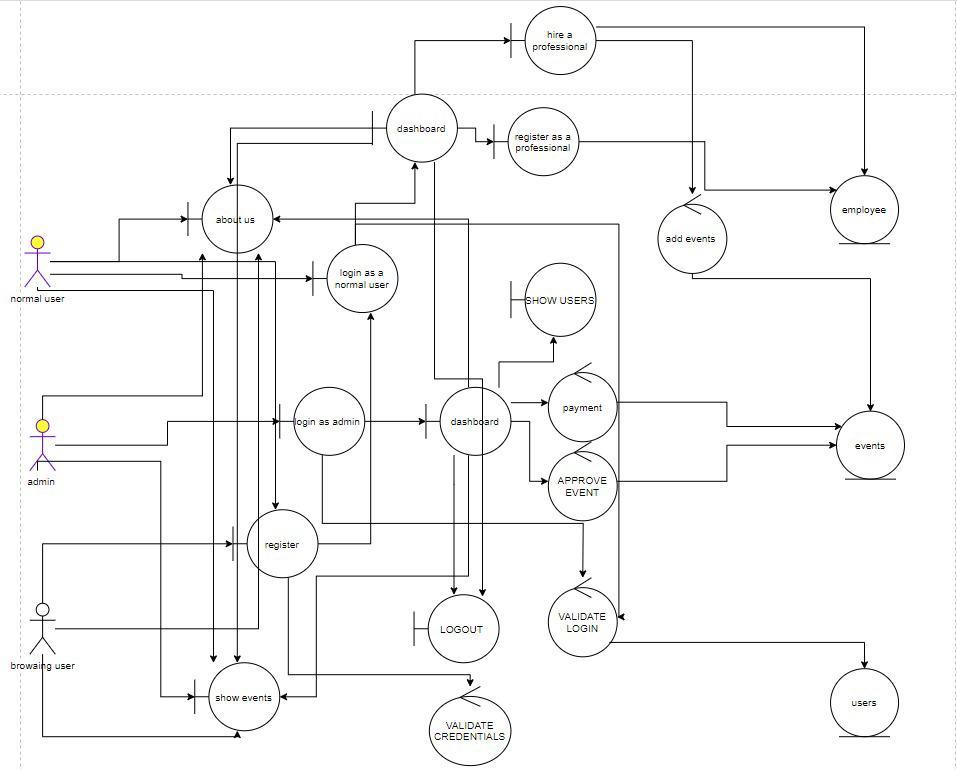
A Normal User will have the access to hire a professional for his event and add an event after hiring the event planner. They can also register as a professional if the user wants to work with the event management team. Both processes will be approved by the admin user of the system.User will also have access to check all the live events that are currently open and book tickets for those events.

* **Browsing User**

A browsing user can also check all the live events that are currently listed on the site and if he wishes to book tickets for an event the system will ask the user to register. The registration is verified by the admin user and after registration the browsing user will be able to book tickets for a listed event.

* **Admin**

The admin user interface will have most of the processing involved. They will be capable of viewing the details of all the users that are registered in the system. If a normal user wants to add an event, it needs to be approved by the admin and after approving the event, the event gets stored in the events database. They also manage the payments of the tickets booked by the user and also approve the payment normal users have to pay after adding an event.



## 

## **ACTIVITY DIAGRAMS**

We use Activity Diagrams to illustrate the flow of control in a system and refer to the steps involved in the execution of a use case. An activity diagram focuses on the condition of flow and the sequence in which it happens.

* **Login**

In this login activity diagram we have illustrated the steps involved in execution of the use case (login) where two actors are involved in the process (User , System). Here the user has two options, he can choose to either login by putting his details or go to forget the password if he doesn’t remember the password.If he (user) chooses to login he enters details which are verified by system through database, system verifies email and its associated password concurrently and logs user in if details are correct, if details are incorrect system maintains a count for wrong inputs, system blocks user after 2 failed attempts .If user chooses forget password option, user has to input email through which system retrieves its associated phone number from database so it can send an OTP , system has an interrupt region and an interrupt occurs on multiple occasion. It occurs if

* User fails to enter code within 30 seconds
* User enters wrong code
* User cancels process in middle

Each time an interrupt occurs a security breach email is sent to the account holder’s email. If the user puts the correct code password is displayed ending our login use case.

* **AddEvent**

In this add event activity diagram we have illustrated the steps involved in execution of use cases (add event) where three actors are involved in the process (User, System and Admin). User adds event details, the system verifies details and if details are valid it adds them to the pending table. Admin approves or rejects the event, system moves event to events table or just removes it from pending table respectively. Email is sent to the host in either condition specifying the state of the event.

## C:\Users\Dell\AppData\Local\Microsoft\Windows\INetCache\Content.Word\LoginActivity.vpd.png

## **SEQUENCE DIAGRAM**

A sequence diagram simply depicts interaction between objects in a sequential order i.e. the order in which these interactions take place. Sequence diagrams describe how and in what order the objects in a system functions.

* **Add Event**

This sequence diagram shows how 6 actors involved in the process interact during the add event use case. Here the use case starts with the user invoking addEvent () function from events class and adds an event which then system adds to the pending table in the database. Then admin approves or rejects the event, in sequence then there is an alt fragment in which the system checks if event is accepted or rejected does some functions depending on the guard condition. In rejection case event is just removed from pending table and system notifies user about the rejection while on other hand in acceptance case event is just removed from pending table, added to events table and system notifies user about the acceptance. A browsing user can only look for an event once it is uploaded, the user pays for a ticket and systems check the validity of payment , system sends ticket booked or invalid ticket maintained in an alt fragment depending on the guard condition. System created payment object is payment is valid hence ending our sequence diagram.

* **Login**

This sequence diagrams show the sequence of messages passed between objects in execution of use case (login). Sequence starts with user entering details and pressing login button on loginUI , loginUI sends user details to system which system verifies through database, next in sequence diagram is an alt fragment where system logs in the user and redirects it to user dashboard or system rejects login and send wrong details message on screen depending on what guard condition is.

### **ADD EVENT**C:\Users\Dell\AppData\Local\Microsoft\Windows\INetCache\Content.Word\SequenceDiagramAddEvent.vpd.png

### 

### 

### **LOGIN**

### C:\Users\Dell\AppData\Local\Microsoft\Windows\INetCache\Content.Word\SequenceDiagramLogin.png

## 

## 

## **COLLABORATION DIAGRAM**

A collaboration diagram, also known as a communication diagram, is an illustration of the relationships and interactions among software in the Unified Modeling Language. These diagrams can be used to portray the dynamic behavior of a particular use case and define the role of each object.

* **Login**

This collaboration diagram illustrates the workflow of use case (login) where 3 objects are involved and they interact through message transfers, at first user sends username to system which is verified by system through database, second user enters password and sends it to system which is verified by system through database, system returns login message by logging the user in hence ending our use case.

* **Add Event**

These two collaboration diagrams show the alternate workflows of use cases (Add event) depending on what the admin does. Here 4 objects are involved in the process. Use case starts with a user invoking addEvent () function from events class and adds an event which then adds to the pending table. Admin invokes look for event() from event class and looks for pending events, Admin rejects or approves the event ,In rejection case system just removes event from pending table and event deleted message is returned while on the other hand if admin approves the event , system removes event from pending table and adds it to events table sending back acceptance notification to user.

### 

### 

### 

### 

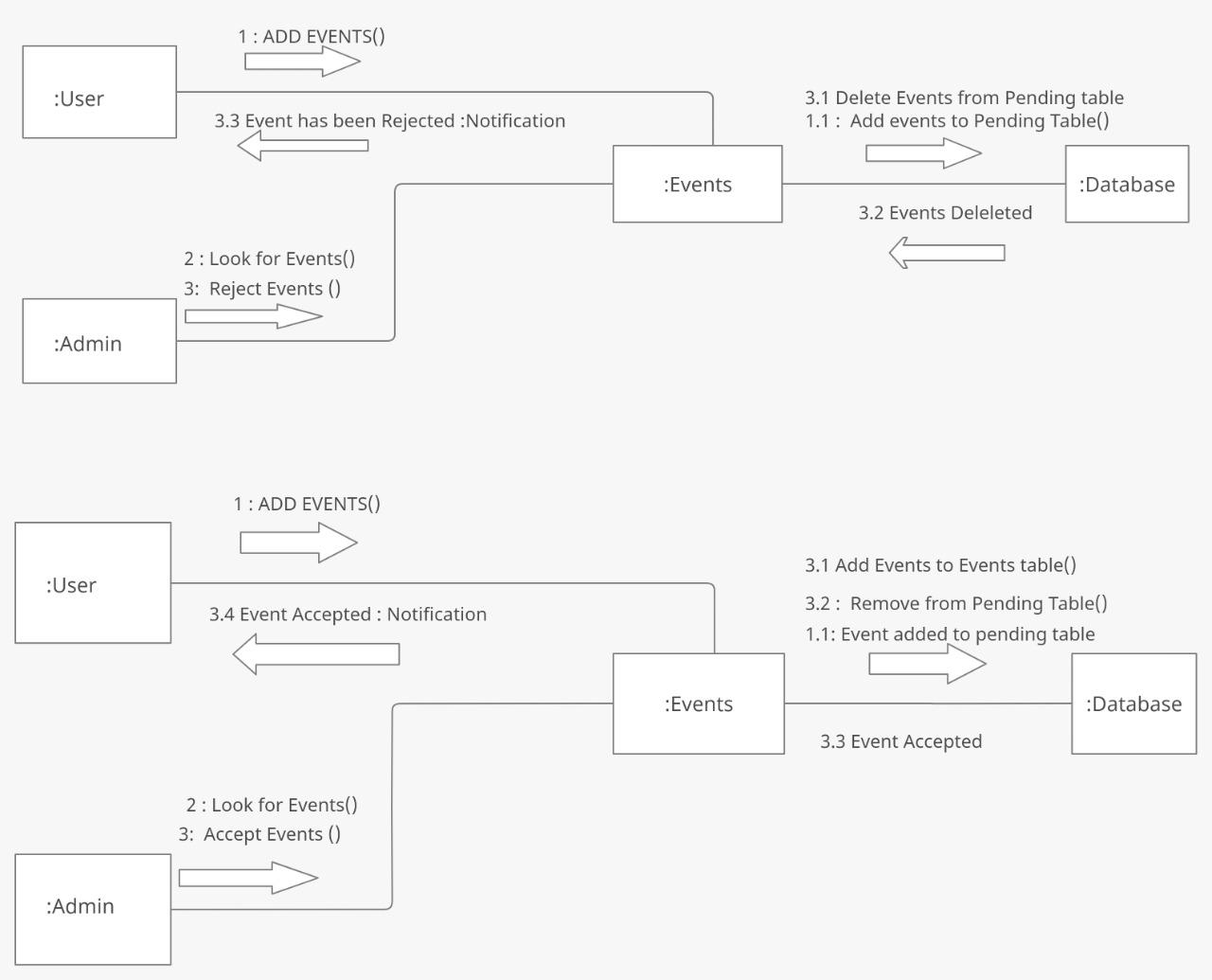
### 

### 

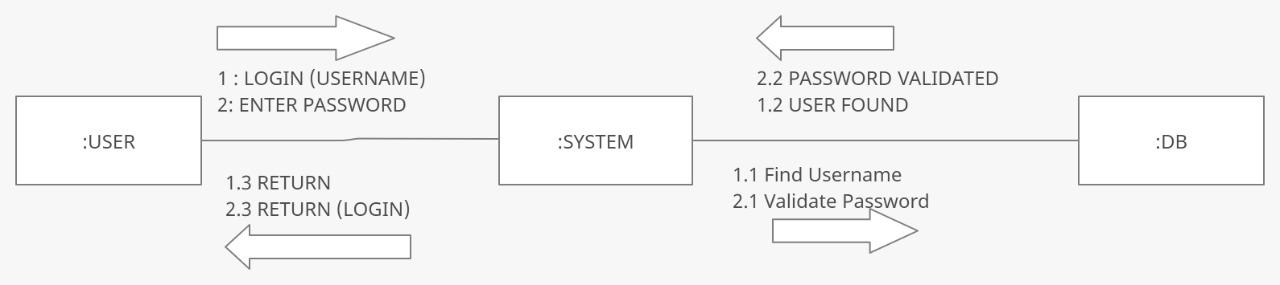
### 

### 

### **ADD EVENT**



### **LOGIN**



## 

## **STATE TRANSITION DIAGRAM**

State transition diagrams describe all of the states that an object can have, the events under which an object changes state (transitions).There are 2 types of state-transition diagrams. We can either show states of an object or web page navigation

* **Web Page**

This diagram shows how our website changes its state by navigating through web pages. All they do is elaborate what our website would do staying at that same particular page (state). Diagram starts when the user enters the website URL (triggering event) and the website renders Home Page. User navigates through web pages (states) by clicking buttons (triggering events). Some states can only be reached through a particular state, e.g. a user can only get to a pending events state if he is on Admin dashboard state.

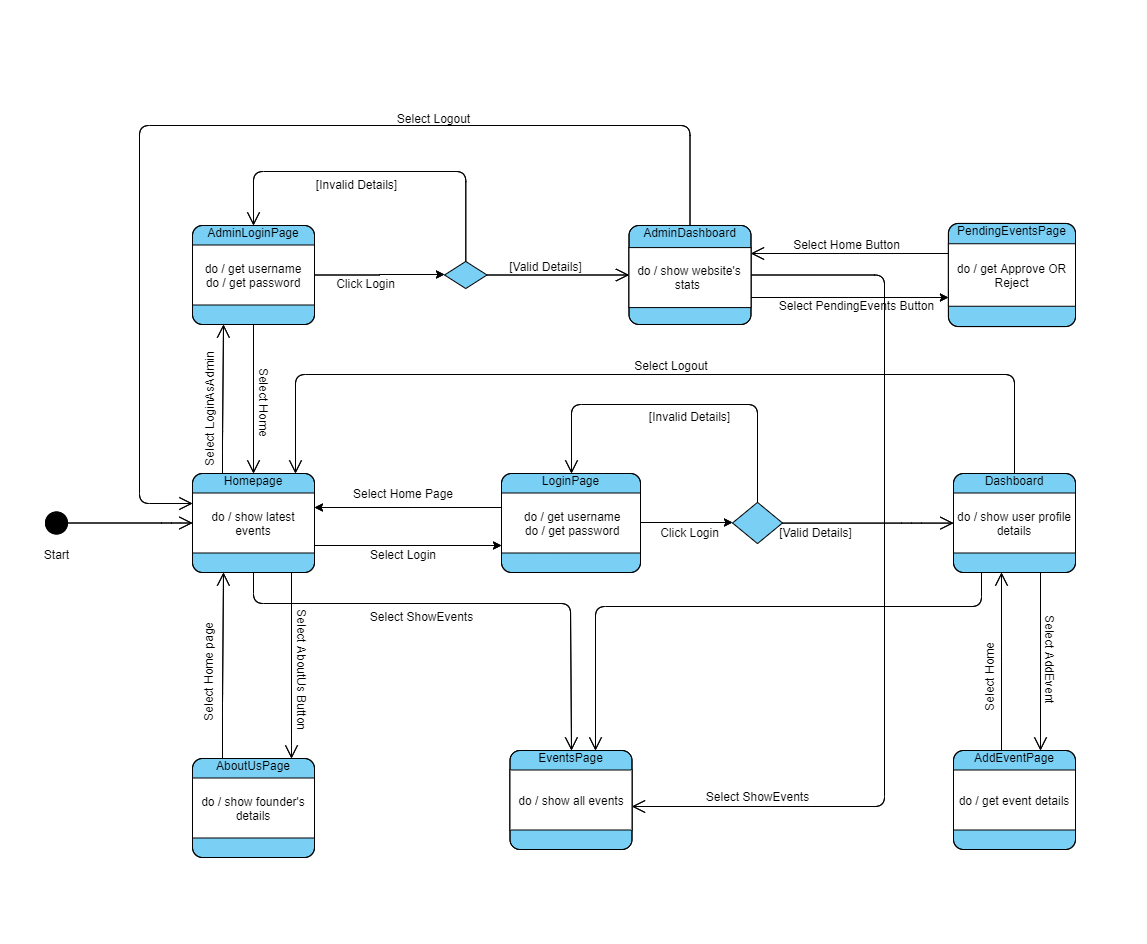
* **User Object**

This diagram shows the states a user object could be during the lifetime of our website, User is in active state once it is created, and during active state user can either be in logged in state or logged out state. Users automatically get suspended if it is inactive for a year or admin bans it. Users can get back to active state if the admin unbans it. Admin can also delete a user changing user’s state to deleted state.

* **Event Object**

This diagram shows the states an event object could be during the lifetime of our website, User is in pending, approved or deleted state. Event is in pending state once it is added by the user. Admin either approves or rejects the event, if approved event state changes to approved (active) state, if rejected event state changes to deleted state. Event would also transit automatically from active to deleted state if currentDate gets larger than eventDate.

### **STATE DIAGRAM #1**



### **STATE DIAGRAM # 2**C:\Users\Dell\AppData\Local\Microsoft\Windows\INetCache\Content.Word\UserObjectStateMachineDiagram.png**STATE DIAGRAM # 3**C:\Users\Dell\AppData\Local\Microsoft\Windows\INetCache\Content.Word\eventobjectstatemachine.png

## **DEPLOYMENT DIAGRAM**

Deployment diagrams are typically used to visualize the physical hardware and software of a system. Using it you can understand how the system will be physically deployed on the hardware.

* **Nodes**
  + A **web server** is the software that receives your request to access a web page. It runs a few security checks on your HTTP request and takes you to the web page. Depending on the page you have requested, the page may ask the server to run a few extra modules while generating the document to serve you. It then serves you the document you requested so our main program file go there.
  + **Database servers** are used to store and manage databases that are stored on the server and to provide data access for authorized users so our database files goes there.
* **Connections used**
  + **Hypertext Transfer Protocol** (HTTP) is an [application-layer](https://en.wikipedia.org/wiki/Application_Layer) protocol for transmitting hypermedia documents, such as HTML .It was designed for communication between web browsers and web servers.
  + An **ODBC** driver uses the Open Database Connectivity (ODBC) interface by Microsoft that allows applications to access data in database management systems (DBMS) using SQL as a standard for accessing the data.

