**CSE 5322- Group 4 Scenarios, Scenario Table and Sequence Diagrams**  
  
**Scenarios:**

1. **Create an Online Account**
   1. The User provides their email address and the account type and clicks on the submit button on the CreateAccount GUI.
   2. The CreateAccount GUI sends the User data to the CreateAccount Controller.
   3. The CreateAccount controller maps the User data to an User Object.
   4. The CreateAccount controller sends a request to DBManager to check if User exists in DB, if not creates the new User Object into the DB using a DBManager.
   5. The CreateAccount controller generates a successful registration message based on the successful response from DBManager.
   6. The CreateAccount controller sends the successful registration message to the CreateAccount GUI.
   7. The CreateAccount controller generates a temporary password and sends it to the user via email.
   8. The CreateAccount GUI displays the successful registration message to the user.
2. **Login**
   1. The user provides their email address and password and clicks on the Login button on the UserLogin GUI.
   2. The UserLogin GUI sends the user data to the UserLogin Controller.
   3. The UserLogin Controller maps the user data to the User Object.
   4. The UserLogin Controller sends the User Object to the DBMgr to verify whether existing LoginInfo is present in the DB.
   5. The DbMgr verifies the User Object data and sends a response to the UserLogin Controller.
   6. If existing LoginInfo is not available, then the UserLogin Controller redirects to the ChangePassword GUI. If existing LoginInfo is available, then the UserLogin Controller will generate a successful Login message.
   7. The UserLogin Controller sends the successful Login message to the UserLogin GUI.
   8. The UserLogin GUI displays the successful Login message to the user.
3. **Submit an Event Proposal**
   1. The Event Organizer provides the Event Proposal information and clicks on the submit button on the submitEvent GUI.
   2. The submitEvent GUI sends the Event Proposal data to the submitEvent Controller.
   3. The submitEvent Controller maps the event data to a new Event Object.
   4. The submitEvent Controller saves the new Event Object to the DB using a DB Manager.
   5. The submitEvent Controller directs to the paymentInfo GUI.
   6. The EventOrganizer is able to select the payment type and submit the payment on the paymentInfo GUI.
   7. The paymentInfo GUI sends the payment data to the paymentInfo Controller.
   8. The paymentInfo Controller maps the payment data to a new Payment Object.
   9. The paymentInfo Controller saves the payment Object to the DB using a DB Manager.
   10. The paymentInfo Controller generates a successful payment message based on the successful response from DBManager.
   11. The paymentInfo controller sends the successful payment message to the submitEvent GUI.
   12. The submitEvent GUI displays the successful proposal submission message to the Event Organizer.
4. **Register for an Event**
   1. The participant provides the information required for the event registration and clicks on submit button on the EventRegistration GUI.
   2. The EventRegistration GUI sends the registration data to the EventRegistration Controller.
   3. The EventRegistration Controller maps the registration data to the specific User Object.
   4. The EventRegistration Controller saves the updated User Object to the DB using a DB Manager.
   5. The EventRegistration Controller directs to the paymentInfo GUI.
   6. The Participant is able to select the payment type and submit the payment on the paymentInfo GUI.
   7. The paymentInfo GUI sends the payment data to the paymentInfo Controller.
   8. The paymentInfo Controller maps the payment data to a new Payment Object.
   9. The paymentInfo Controller saves the payment Object to the DB using a DBManager.
   10. The paymentInfo Controller generates a successful payment message based on the successful response from DBManager.
   11. The paymentInfo controller sends the successful payment message to the EventRegistration GUI.
   12. The EventRegistration GUI displays the successful event registration message to the participant.

**Scenario Table**

**Use Case 1: Create an Account**

| **#** | **Subject** | **Subject Action** | **Object** | **Object Acted on** |
| --- | --- | --- | --- | --- |
| **3.** | **User** | **provides** | **Email address/Account Type** | **CreateAccount GUI** |
| **3.** | **CreateAccount GUI** | **sends** | **User data** | **CreateAccount Controller** |
| **3.** | **CreateAccount Controller** | **maps** | **User data** | **User Object** |
| **3.** | **CreateAccount Controller** | **creates** | **User Object** | **DBManager** |
| **4.** | **CreateAccount**  **Controller** | **generates** |  | **Successful registration message** |
| **4.** | **CreateAccount Controller** | **sends** | **Successful registration message** | **CreateAccount GUI** |
| **4.** | **CreateAccount Controller** | **generates** | **Temporary Password** | **Via Email** |
| **5.** | **CreateAccount GUI** | **displays** | **Successful registration message** | **User** |

**Use Case 2: Login**

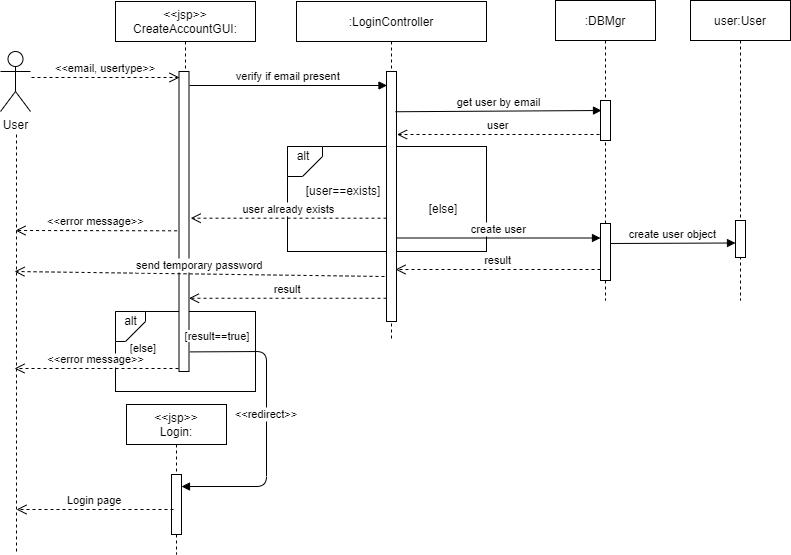
| **#** | **Subject** | **Subject Action** | **Object** | **Object Acted on** |
| --- | --- | --- | --- | --- |
| **3.** | **User** | **provides** | **Email address/Password** | **UserLogin GUI** |
| **3.** | **UserLogin GUI** | **sends** | **User data** | **UserLogin Controller** |
| **4.** | **UserLogin Controller** | **maps** | **User data** | **User Object** |
| **4.** | **UserLogin Controller** | **sends** | **User Object** | **DBManager** |
| **4.** | **DbManager** | **verifies** | **User Object** | **UserLogin Controller** |
| **4.** | **UserLogin Controller** | **redirects** | **LoginInfo** | **ChangePassword GUI** |
| **4.** | **UserLogin Controller** | **generates** |  | **Successful Login message** |
| **4.** | **UserLogin Controller** | **sends** | **Successful Login message** | **UserLogin GUI** |
| **5.** | **UserLogin GUI** | **displays** | **Successful Login message** | **User** |

**Use Case 3: Submit an Event Proposal**

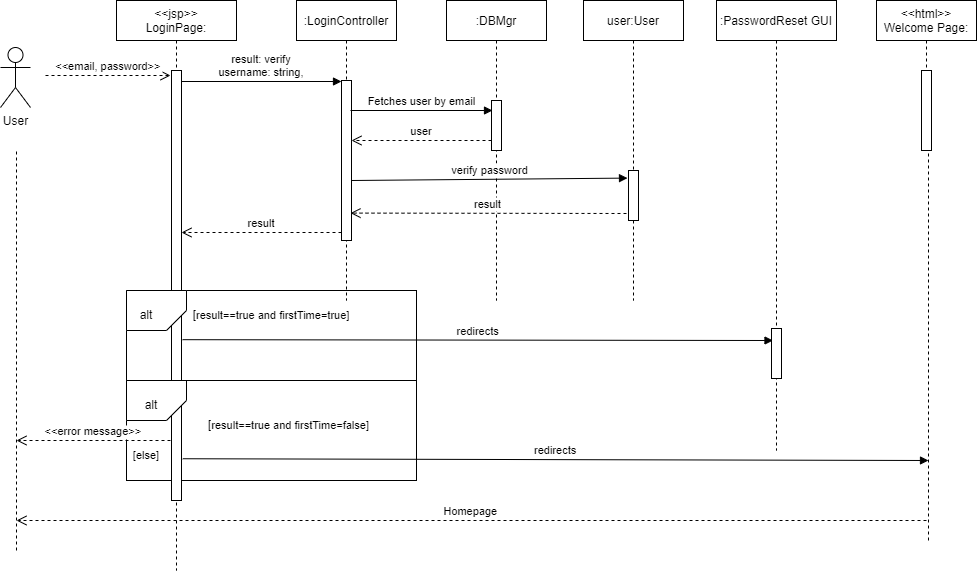
| **#** | **Subject** | **Subject Action** | **Object** | **Object Acted on** |
| --- | --- | --- | --- | --- |
| **3.** | **Event Organizer** | **provides** | **Event Proposal Information** | **submitEvent GUI** |
| **3.** | **submitEvent GUI** | **sends** | **Event Proposal data** | **submitEvent Controller** |
| **4.** | **submitEvent Controller** | **maps** | **Event data** | **Event Object** |
| **4.** | **submitEvent Controller** | **saves** | **Event Object** | **DBManager** |
| **4.** | **submitEvent Controller** | **directs** |  | **paymentInfo GUI** |
| **4.** | **Event Organizer** | **selects/submit** | **Payment type/Payment** | **paymentInfo GUI** |
| **4.** | **paymentInfo GUI** | **sends** | **Payment Data** | **paymentInfo Controller** |
| **4.** | **paymentInfo Controller** | **maps** | **Payment Data** | **Payment Object** |
| **4.** | **paymentInfo Controller** | **saves** | **Payment Object** | **DBManager** |
| **4.** | **paymentInfo Controller** | **generates** |  | **Successful Login message** |
| **4.** | **paymentInfo Controller** | **sends** | **Successful Payment message** | **submitEvent GUI** |
| **5.** | **submitEvent GUI** | **displays** | **Successful Payment message** | **Event Organizer** |

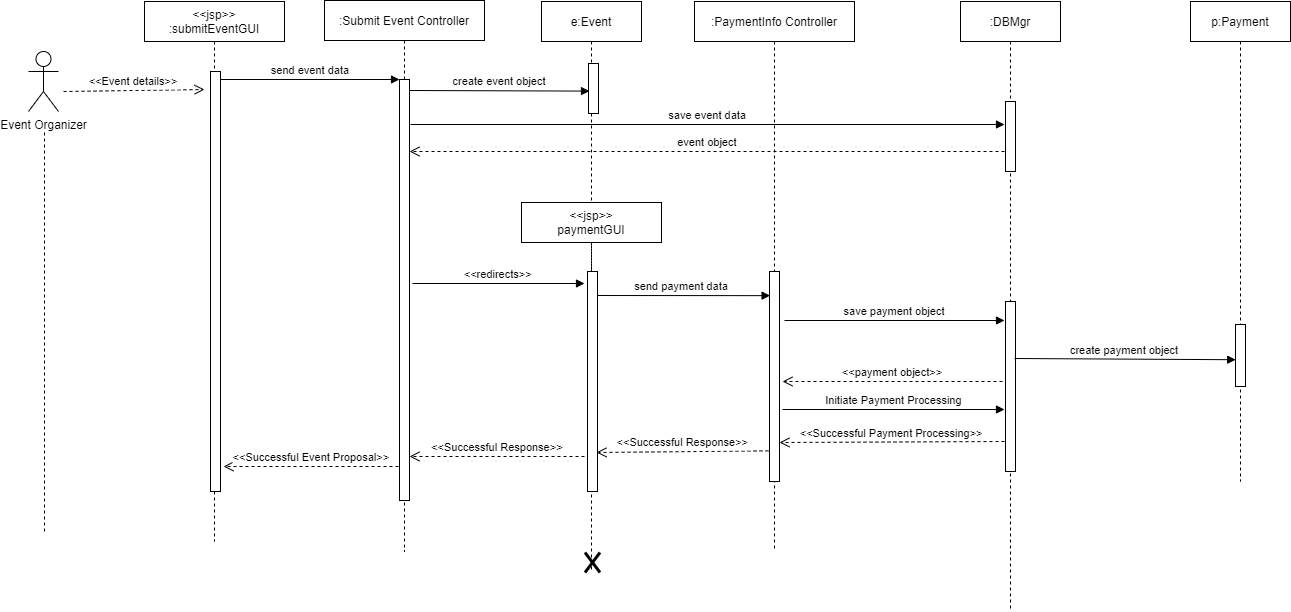
**Use Case 4: Register for an Event**

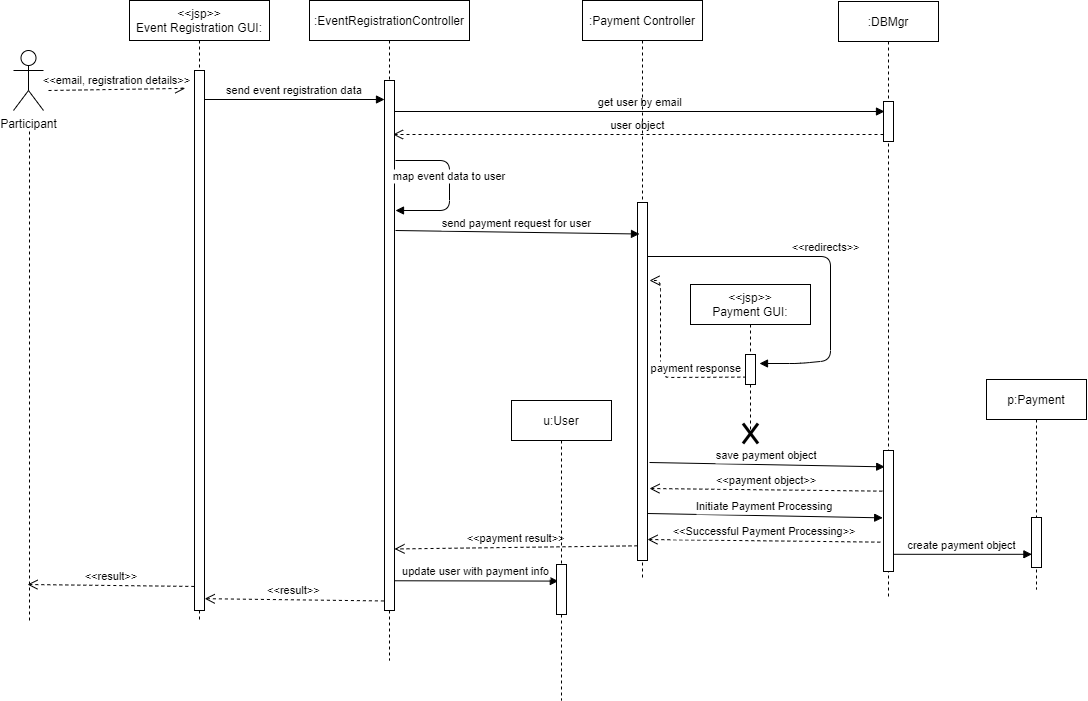
| **#** | **Subject** | **Subject Action** | **Object** | **Object Acted on** |
| --- | --- | --- | --- | --- |
| **3.** | **Participant** | **provides** | **Event Registration Information** | **EventRegistration GUI** |
| **3.** | **EventRegistration GUI** | **sends** | **Registration data** | **EventRegistration Controller** |
| **4.** | **EventRegistration Controller** | **maps** | **Registration data** | **User Object** |
| **4.** | **EventRegistration Controller** | **saves** | **User Object** | **DBManager** |
| **4.** | **EventRegistration Controller** | **directs** |  | **paymentInfo GUI** |
| **3.** | **Participant** | **selects/submit** | **Payment type/Payment** | **paymentInfo GUI** |
| **4.** | **paymentInfo GUI** | **sends** | **Payment Data** | **paymentInfo Controller** |
| **4.** | **paymentInfo Controller** | **maps** | **Payment Data** | **Payment Object** |
| **4.** | **paymentInfo Controller** | **saves** | **Payment Object** | **DBManager** |
| **4.** | **paymentInfo Controller** | **generates** |  | **Successful Login message** |
| **4.** | **paymentInfo Controller** | **sends** | **Successful Payment message** | **eventRegistration GUI** |
| **5.** | **eventRegistration GUI** | **displays** | **Successful Payment message** | **Participant** |

**Informal Sequence Diagrams:  
  
1. Create Account:  
  
**

**2. User Login:**

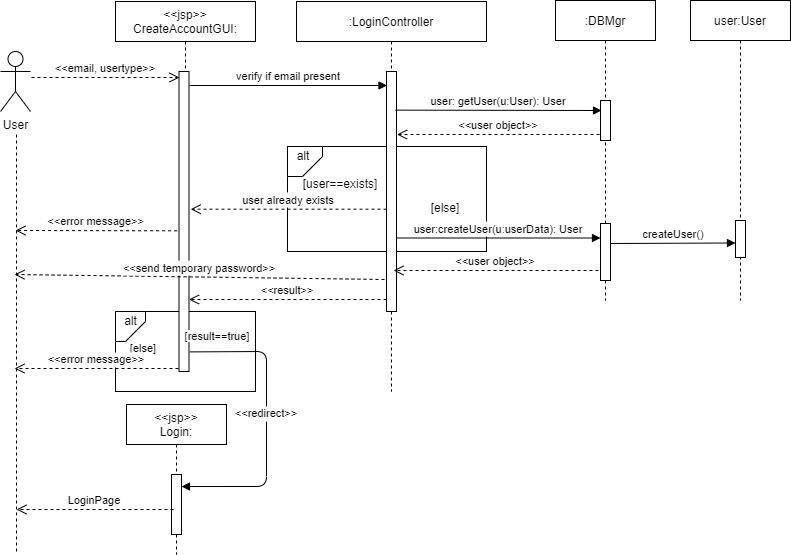
****

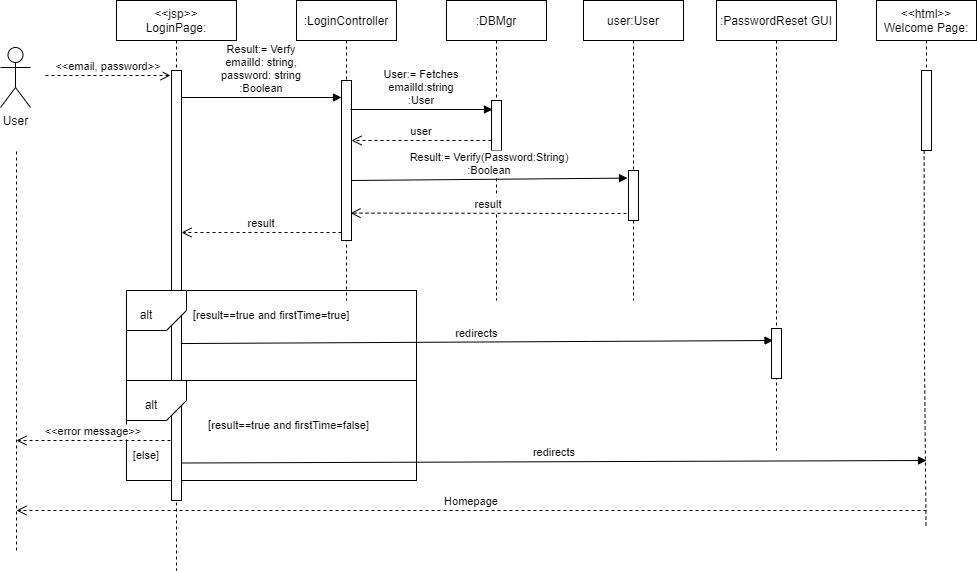
**3. Submit an Event Proposal:  
**

**4. Register for an Event:  
  
**

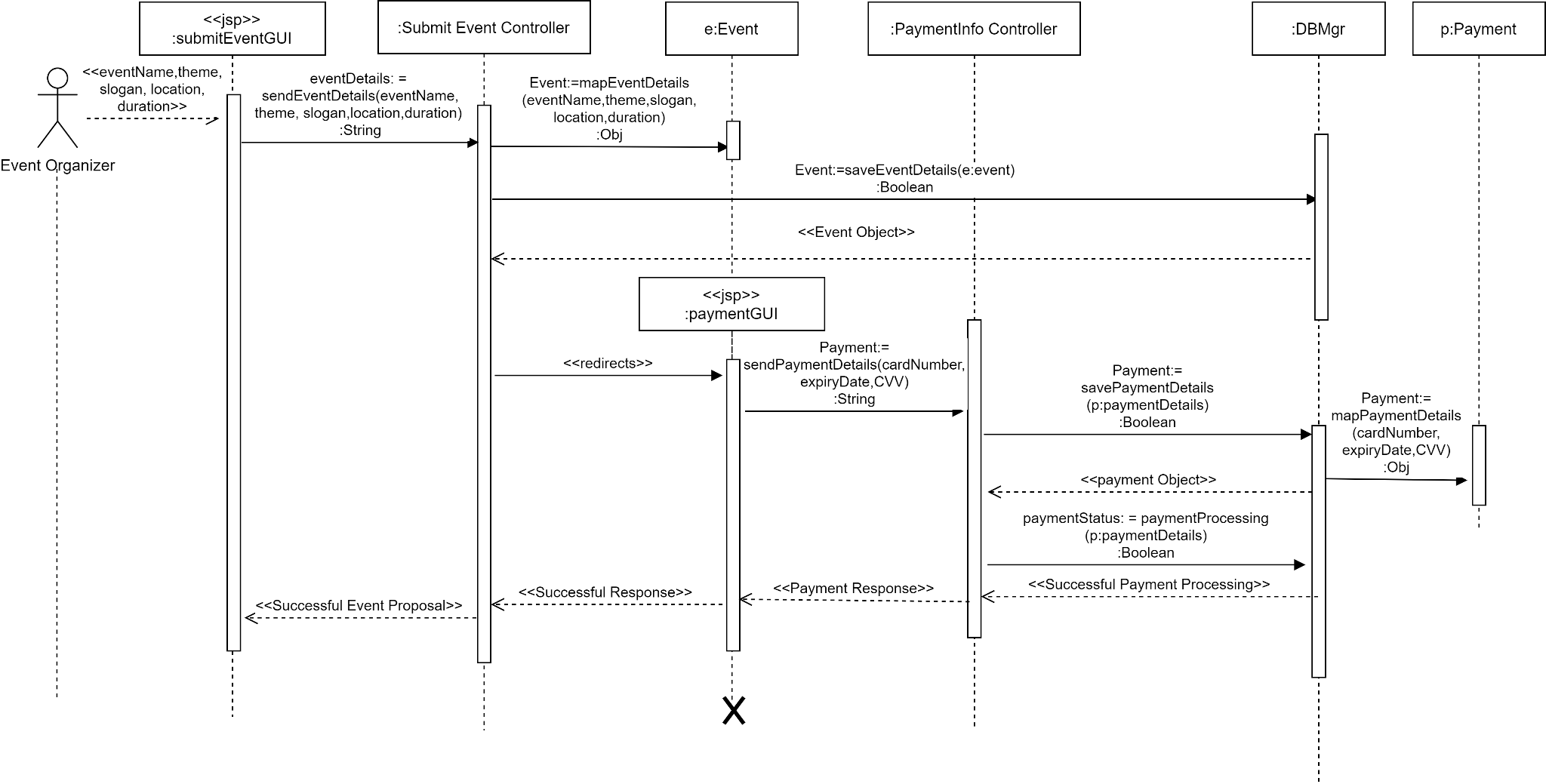
**Design Sequence Diagrams:**

**1. Create Account:**

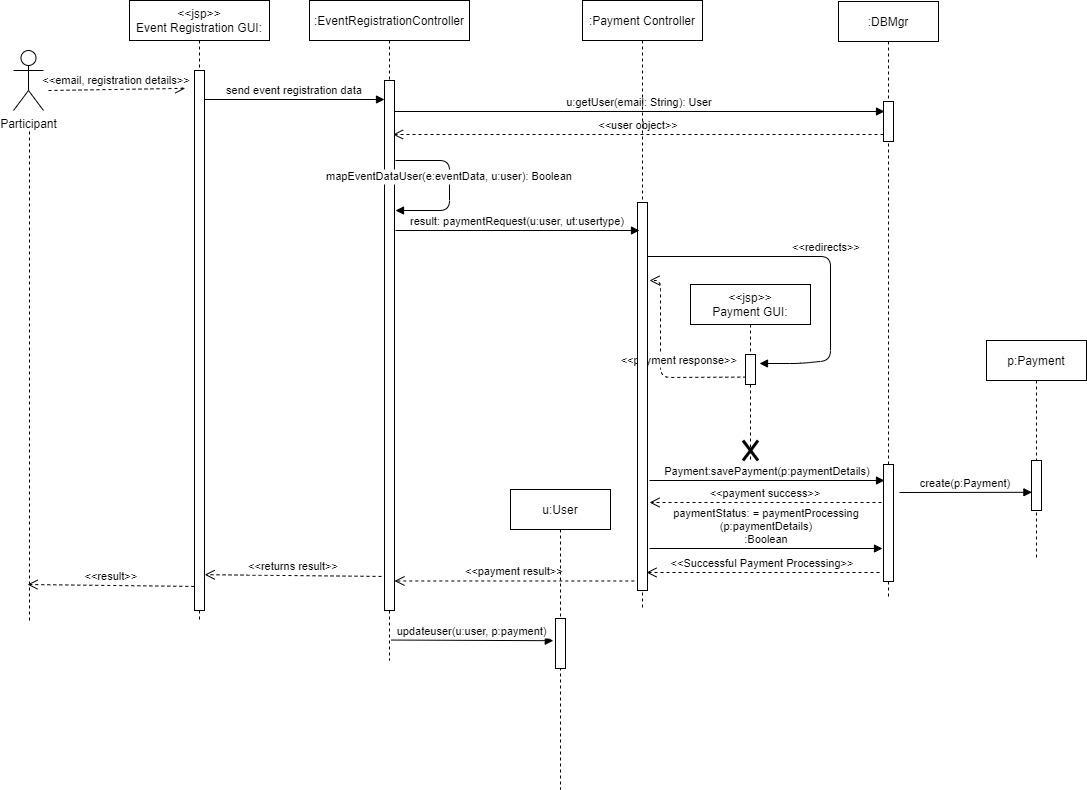
****

**2. Login:  
  
**

**3. Submit an Event Proposal:**

****

**4. Register for an Event:**

****