Team Project: *Group 01*

Team Members

Number	Name	Email(s)	CSGitLab Username
TM1	Qiyue Zhu	qiyue.zhu@ucdconnect.ie	@Victoria
TM2	Wenyi Liang	wenyi.liang@ucdconnect.ie	@Wenyi
TM3	Yunfei Xu	yunfei.xu@ucdconnect.ie	@Kira210
TM4	Zitong Wang	zitong.wang@ucdconnect.ie	@sukikatte
TM5	Siqi Du	siqi.du@ucdconnect.ie	@Yuki
TM6	Jiayi Cai	jiayi.cai@ucdconnect.ie	@22207285
TM7	Yani Yang	yani.yang@ucdconnect.ie	@hs
TM8	Boran Duan	boran.duan@ucdconnect.ie	@boran

Analysis

This phase continues the development process by realising the courses of events that we described in the requirements analysis phase. This requires the creation of sequence diagrams for each course of events. Decisions made while constructing these diagrams are also documented in the class diagram.

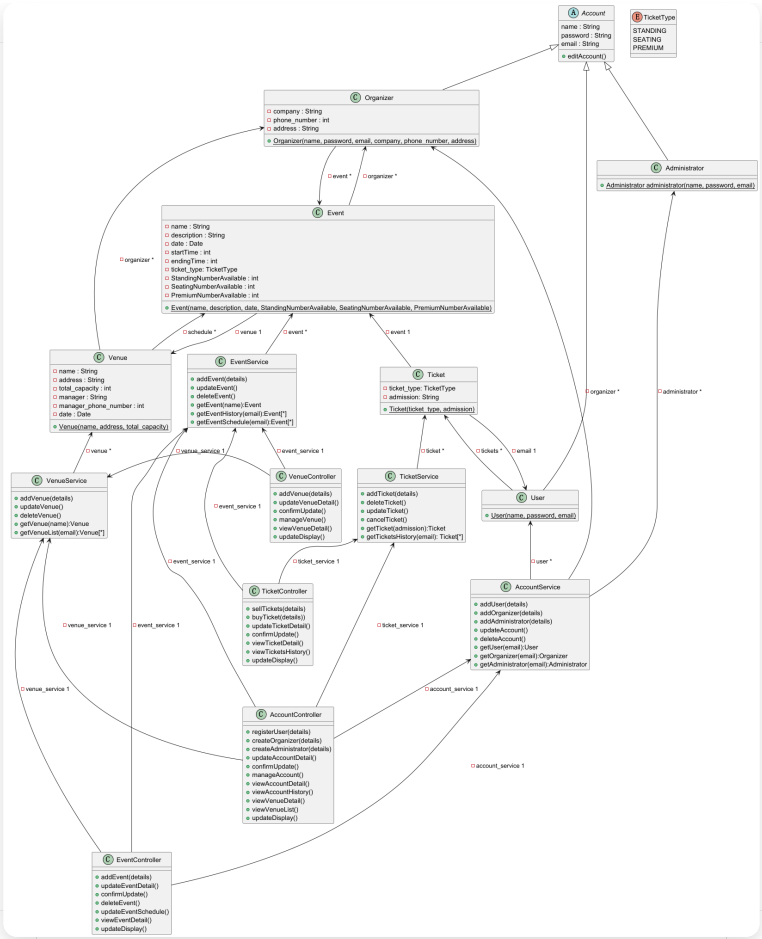
Use Case Realisations (Sequence Diagrams)

Each use case contains the separate sequence diagrams for each course of events. The diagrams are stored in the `images` folder and referenced in the markdown files.

1. Creat Account
2. List Account
3. View Account
4. Modify Account
5. Register Account
6. List Venues
7. Add Venue
8. View Venue
9. Modify Venue
10. Delete Venue
11. List Events
12. Add Event
13. View Event
14. Modify Event
15. Delete Event
16. List My Events
17. List Tickets
18. Buy Tickets
19. View Tickets

Class Diagram

The class diagram represents the information gained about the system by completing the use case realisations.



class diagram

Description of Responsibilities

Account

The `Account` class is an abstract entity class. It represents a user account in the system. The sole responsibility of account objects is to store the account's name, password, and email. To note, the email should be unique for every account.

Administrator

The `Administrator` class is a concrete subclass of the `Account` class. It represents an administrator user in the system. The sole responsibility of administrator objects is to manage system-wide settings and user accounts.

Organizer

The `Organizer` class is a concrete subclass of the `Account` class. It represents an event or a venue organizer in the system. The sole responsibility of organizer objects is to store extra information related to the organizer's company, phone number, and address. Organizers should be responsible for managing the events, venues, tickets that within their rights.

User

The `User` class is a concrete subclass of the `Account` class. It represents a regular user in the system. The sole responsibility of user objects is to manage their tickets for events.

Venue

The `Venue` class is a basic entity class. It represents a physical location where events are held. The sole responsibility of venue objects is to store details about the venue's name, address, total capacity, manager, and manager's phone number. To note, the name should be unique for every venue.

Event

The `Event` class is a basic entity class. It represents an event scheduled at a venue. The sole responsibility of event objects is to store the event's name, description, date, start time, ending time, ticket type, and the number of available tickets (standing, seating, and premium). To note, the name should be unique for every event.

Ticket

The `Ticket` class is a basic entity class. It represents a ticket purchased by a user for an event. The sole responsibility of ticket objects is to store the ticket type and admission details. To note, the admission should be unique for every ticket.

EventController

The `EventController` class is a controller class. It is responsible for managing event-related operations, including adding, updating, confirming updates, deleting events, updating event schedules, and viewing event details.

AccountController

The `AccountController` class is a controller class. It is responsible for managing account-related operations, including registering users, creating organizers and administrators, updating accounts, confirming updates, managing accounts, and viewing account details and histories.

VenueController

The `VenueController` class is a controller class. It is responsible for managing venue-related operations, including adding, updating, confirming updates, managing venues, and viewing venue details.

TicketController

The `TicketController` class is a controller class. It is responsible for managing ticket-related operations, including selling and buying tickets, updating ticket details, confirming updates, viewing ticket details, and viewing ticket history.

VenueService

The `VenueService` class is a service class. It is responsible for performing operations related to venues, including adding, updating, deleting venues, retrieving venue details, and getting the schedule of events at a venue.

EventService

The `EventService` class is a service class. It is responsible for performing operations related to events, including adding, updating, deleting events, and retrieving associated venues and organizers.

AccountService

The `AccountService` class is a service class. It is responsible for performing operations related to accounts, including adding users, organizers, and administrators, updating accounts, deleting accounts, and retrieving account details and histories.

TicketService

The `TicketService` class is a service class. It is responsible for performing operations related to tickets, including adding, deleting, updating, canceling tickets, and retrieving ticket details and histories.

Milestone 2 Analysis

Distribution of work on this milestone

Overall Distribution of Work

Team Member	@Victoria	@Wenyi	@Kira210	@sukikatte	@Yuki	@22207285	@hs	@boran
Percentage	12%	12%	12%	12%	12%	12%	12%	12%

Task Allocation

Item	Contributor 1	Contributor 2	Reviewer
Description of Responsibilities	@Victoria(50%)	@hs(50%)	Other Members
Class Diagram	@Victoria(50%)	@hs(50%)	Other Members
Use Case 1: Create Account	@Kira210(100%)		Other Members
Use Case 2: List Account	@Kira210(100%)		Other Members
Use Case 3: View Account	@Wenyi(100%)		Other Members
Use Case 4: Modify Account	@Wenyi(100%)		Other Members
Use Case 5: Register Account	@Wenyi(100%)		Other Members
Use Case 6: List Venues	@sukikatte（100%）		Other Members
Use Case 7: Add Venue	@boran(100%)		Other Members
Use Case 8: View Venue	@Kira210(100%)		Other Members
Use Case 9: Modify Venue	@boran(100%)		Other Members
Use Case 10: “Delete Venue”	@22207285 (100%)		Other Members
Use Case 11: List Events	@sukikatte（100%）		Other Members
Use Case 12: Add Event	@boran(100%)		Other Members
Use Case 13: “View Event”	@22207285 (100%)		Other Members
Use Case 14: Modify Event	@boran(100%)		Other Members
Use Case 15: “Delete Event”	@22207285 (100%)		Other Members
Use Case 16: List My Event	@sukikatte（100%）		Other Members
Use Case 17: List Tickets	@Yuki（100%）		Other Members
Use Case 18: Buy Tickets	@Yuki（100%）		Other Members
Use Case 19: View Tickets	@Yuki（100%）		Other Members

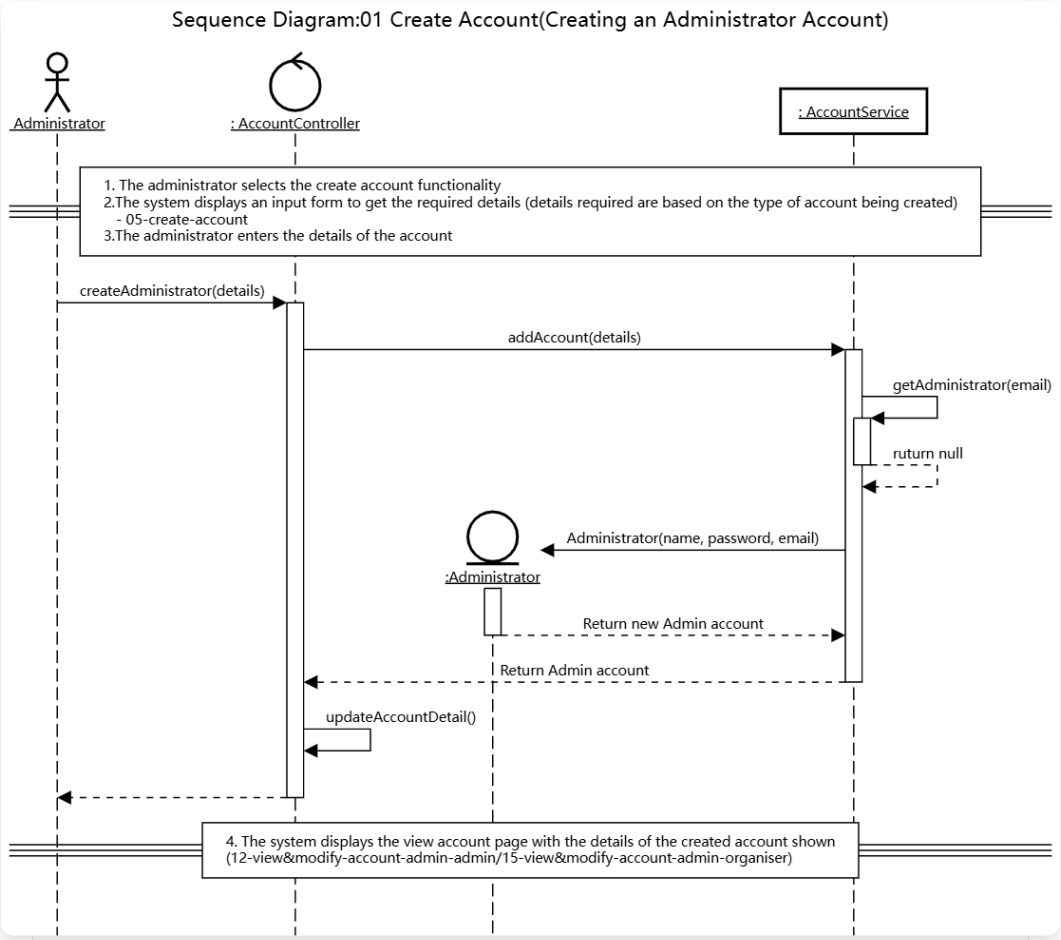
Reflection Statements

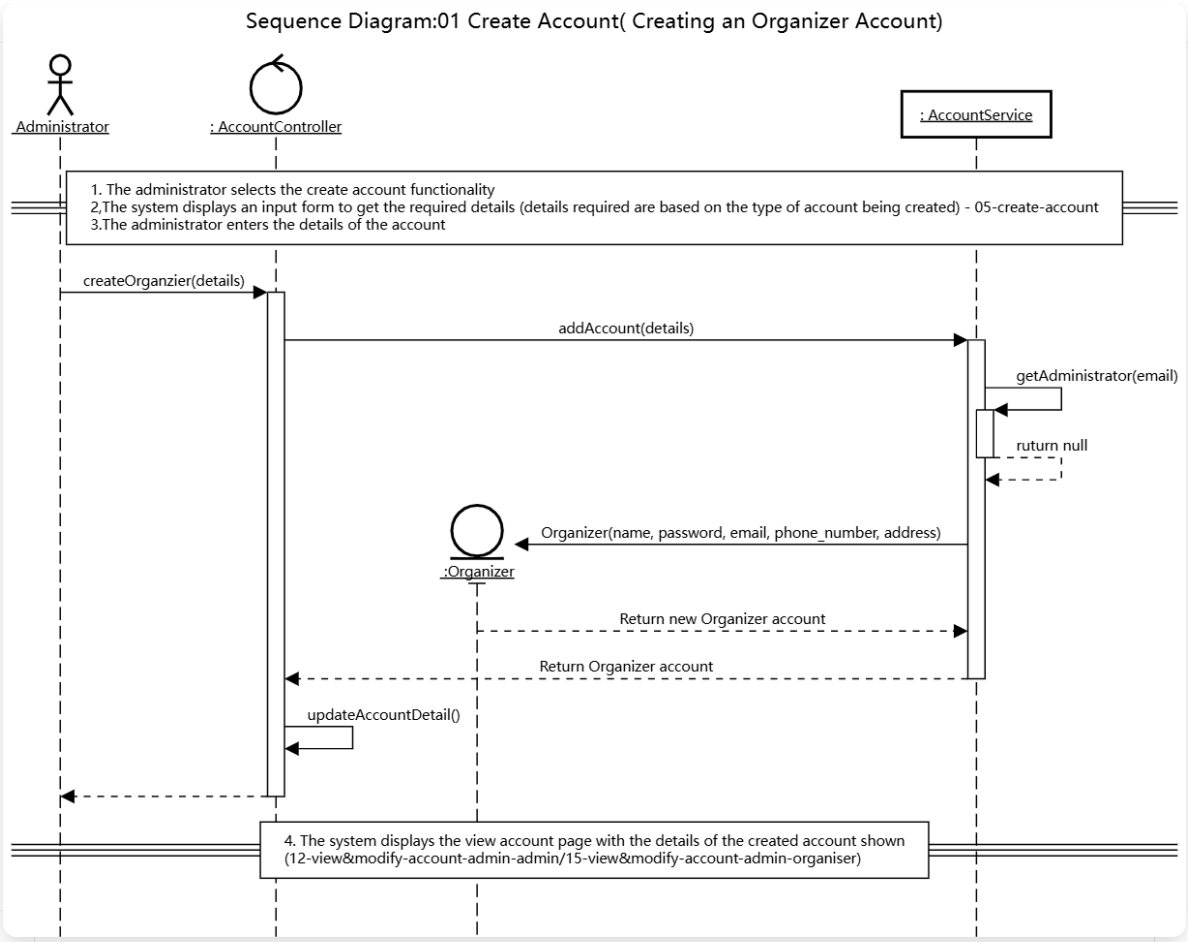
Team Member	Contribution Reflection Statement
@hs	Participated in class diagramming and writing of description of responsibility. Overview other members' usecase.
@Boran	Drew the sequence diagrams and wrote docs of use case 07, 09, 12 and 14 about its basic event and alternate events.
@sukikatte	Drew the sequence diagramS and wrote docs of use case 06, 11 and 16 about its basic event and alternate events.
@Kira210	Drew the sequence diagramS and wrote docs of use case 01, 02 and 08 about its basic event and alternate events.
@Yuki	Drew the sequence diagramS and wrote docs of use case 17, 18 and 19 about its basic event and alternate events.
@Wenyi	Drew the sequence diagrams and wrote docs of use case 03, 04 and 05 about its basic event and alternate events.
@Victoria	Participated in class diagramming, writing and modifying the description of responsibility and modify the domain model diagram accordingly. Reviewing the work of other members and providing suggestions. Following up on the team's progress.
@22207285	Drew the sequence diagrams and wrote docs of use case 10, 13 and 15 about its basic event and alternate events.

Detailed Use Case Analysis

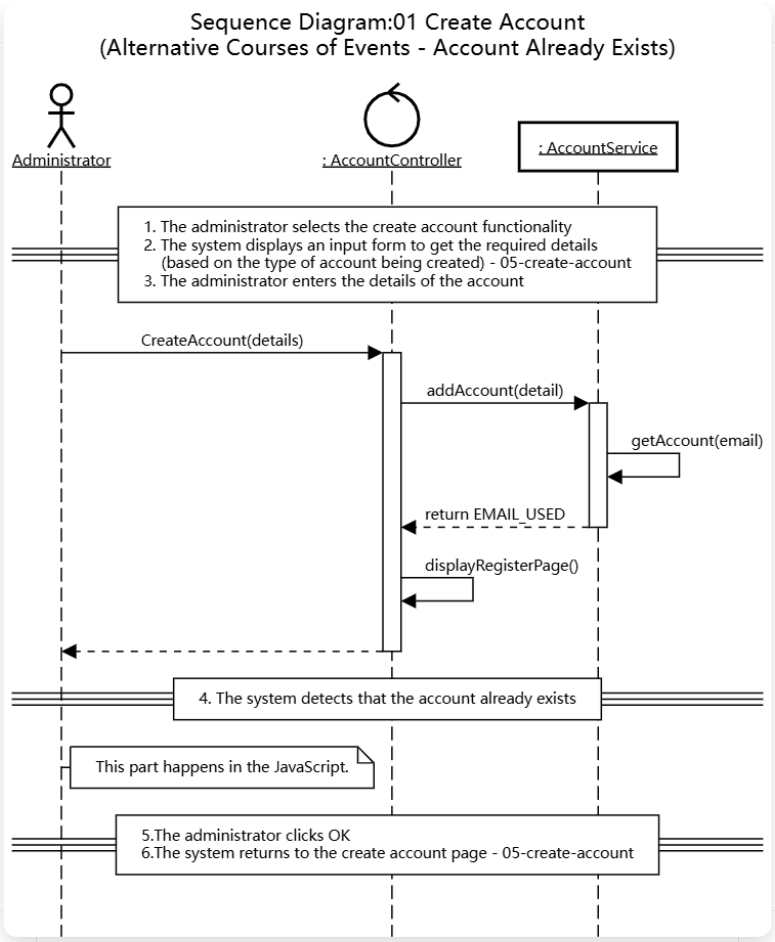
Use Case 01 - Create Account

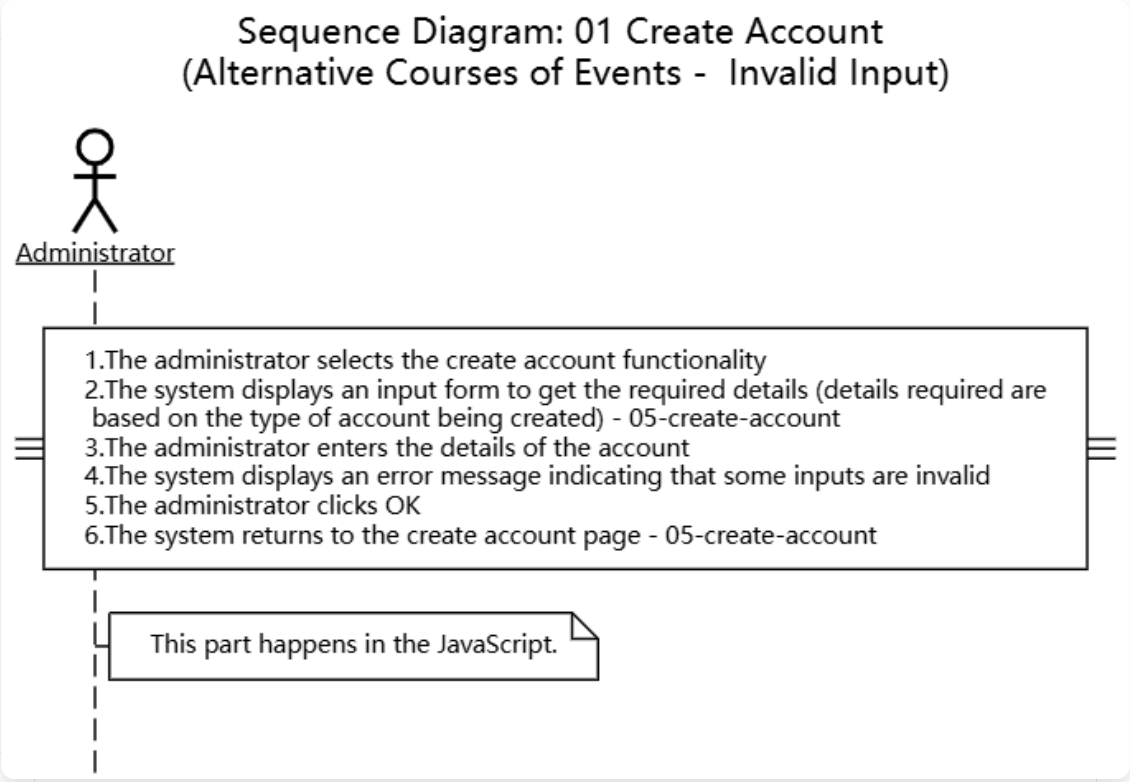
01 - Basic Course of Events



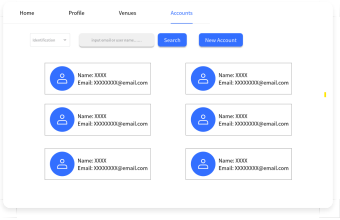
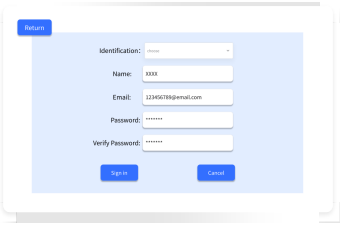
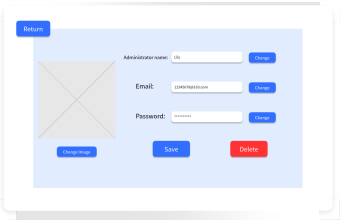
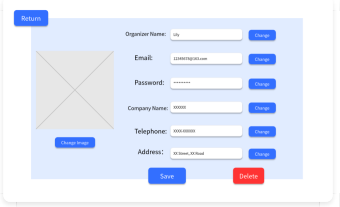


02 - Alternate Course of Events - Some Explanation



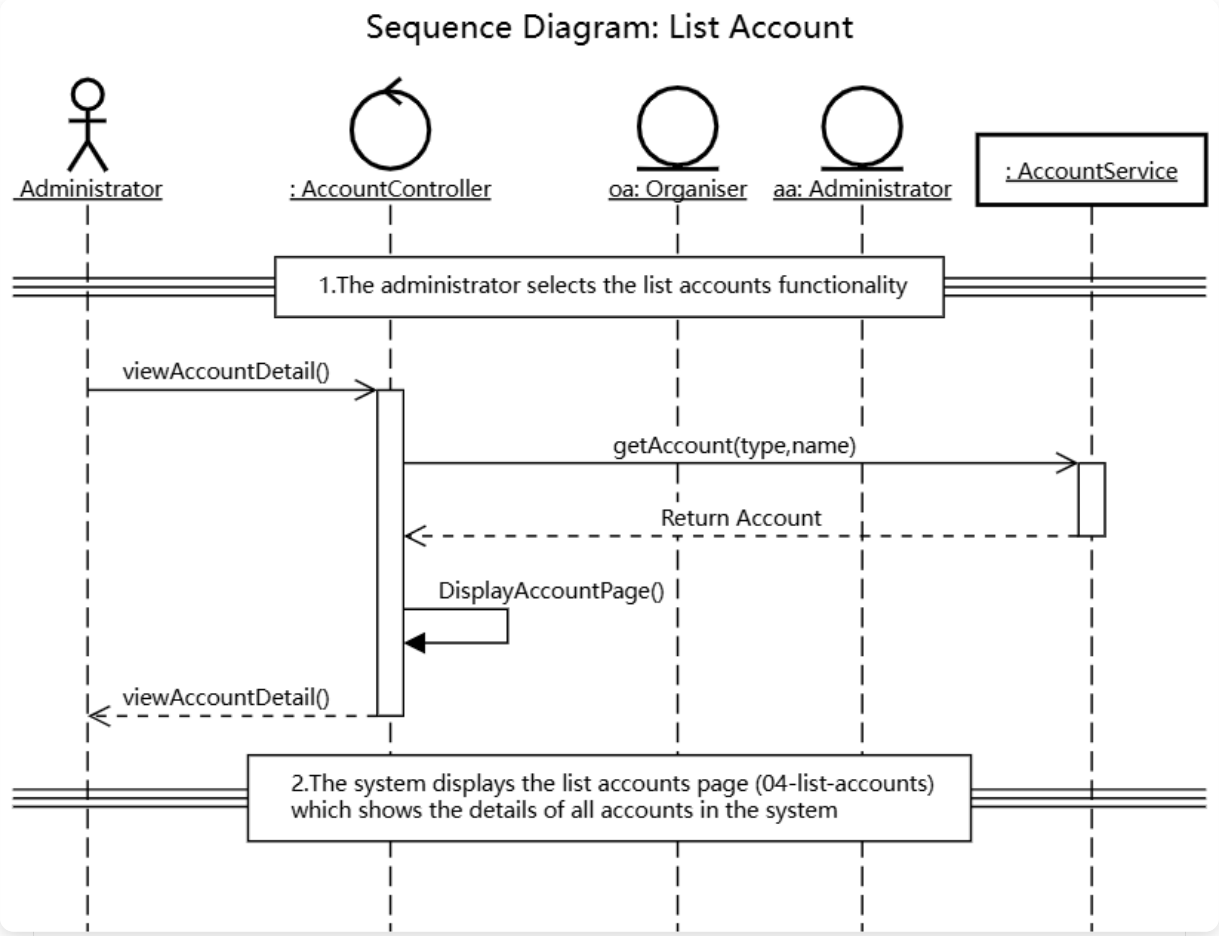


Related UI Prototypes

Page Name	Image
List Accounts Page	
Create Account Page	
View Account Page (Admin-Admin)	
View Account Page (Admin-Organiser)	
Duplicate Email Message	Duplicate Email Message
Missing Details Message	Missing Details Message

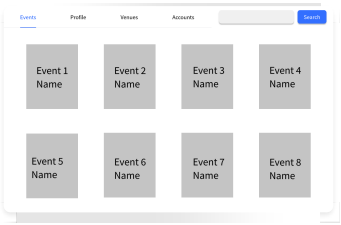
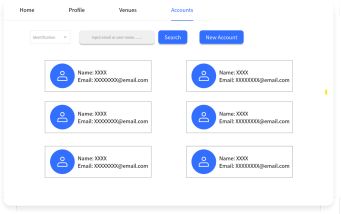
Use Case 02 - List Account

01 - Basic Course of Events



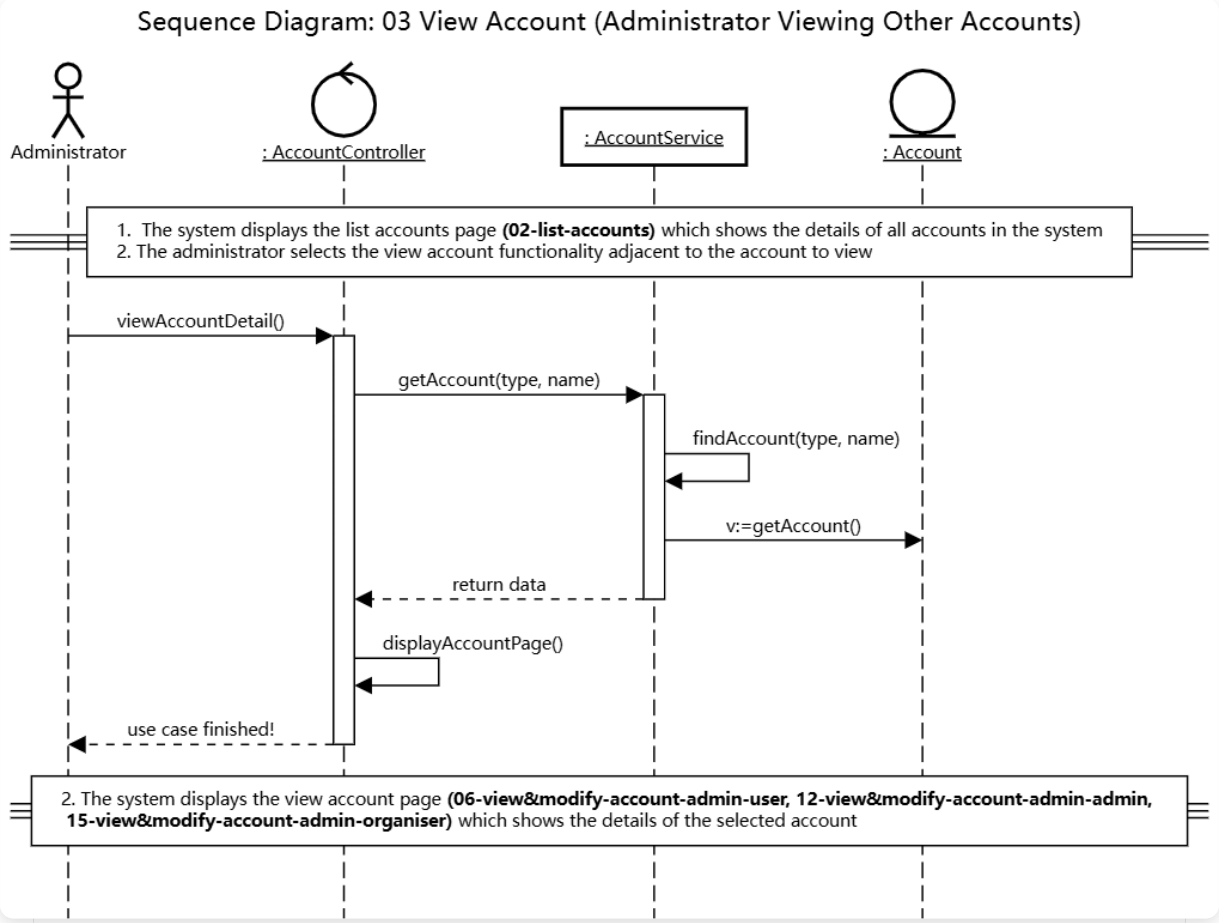
LIst Account - Basic Course of Events

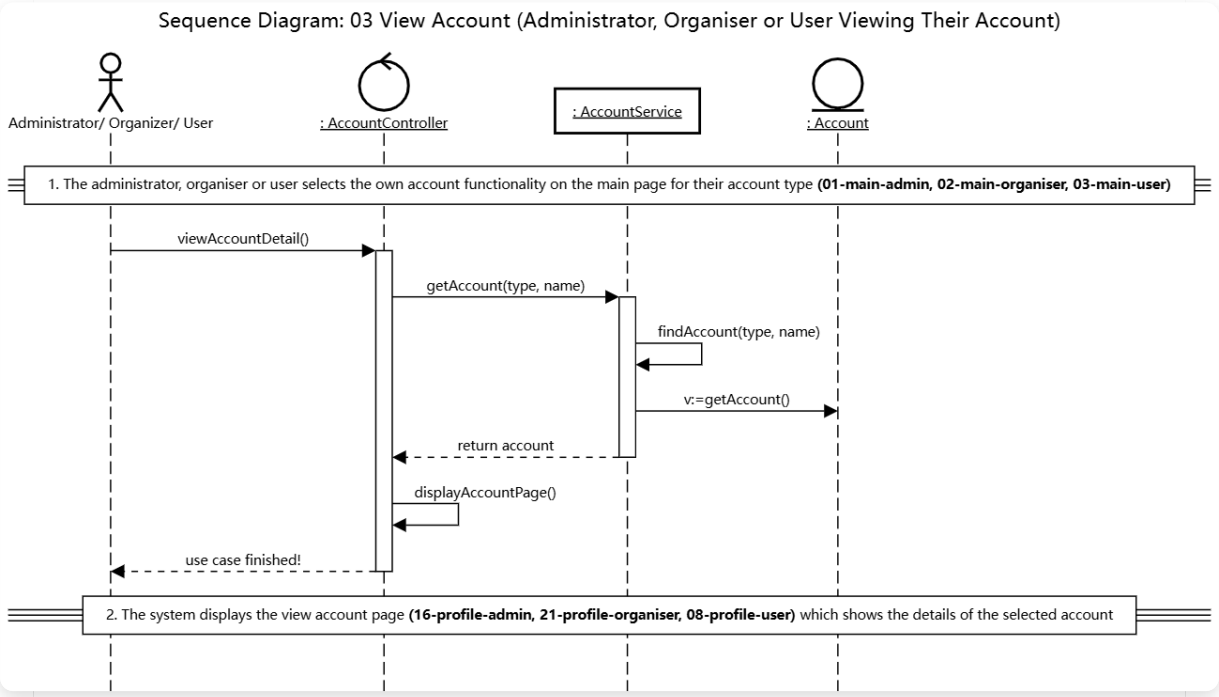
Related UI Prototypes

Page Name	Image
Admin Main Page	
List Accounts Page	

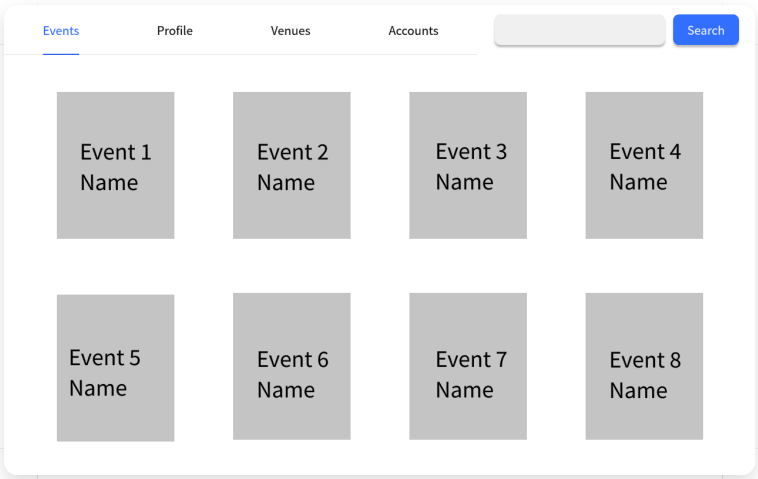
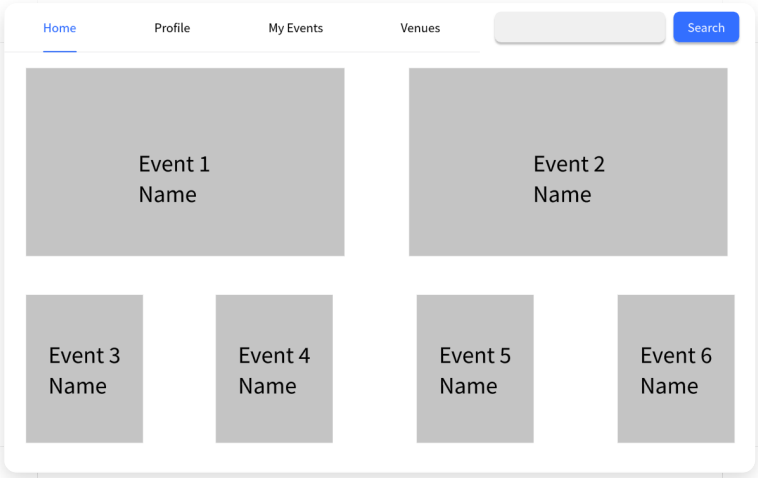
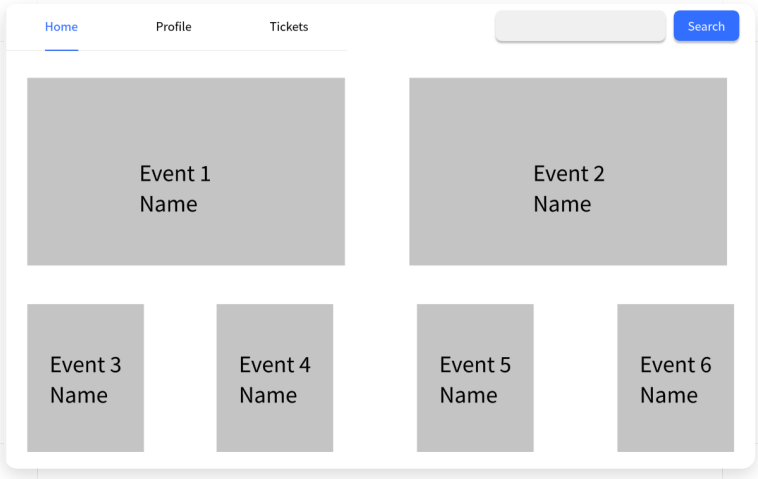
Use Case 03 - View Account

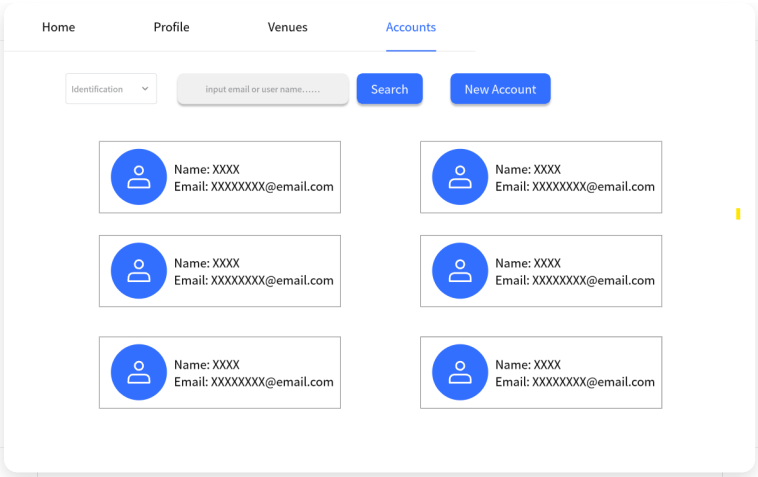
01 - Basic Course of Events



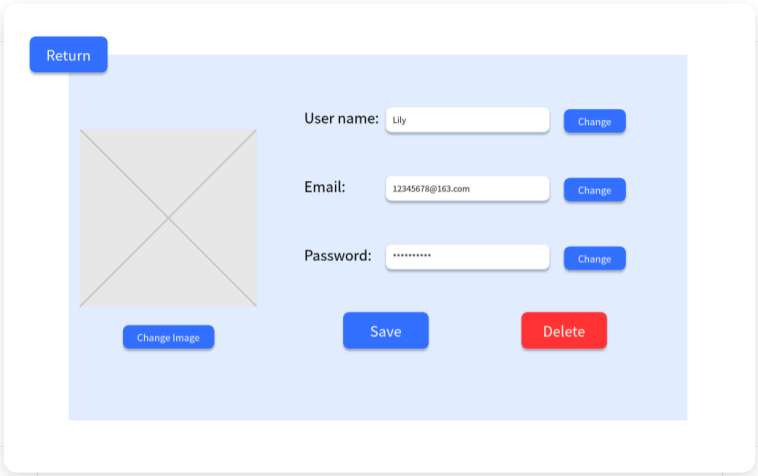


Relevant UI Sketches

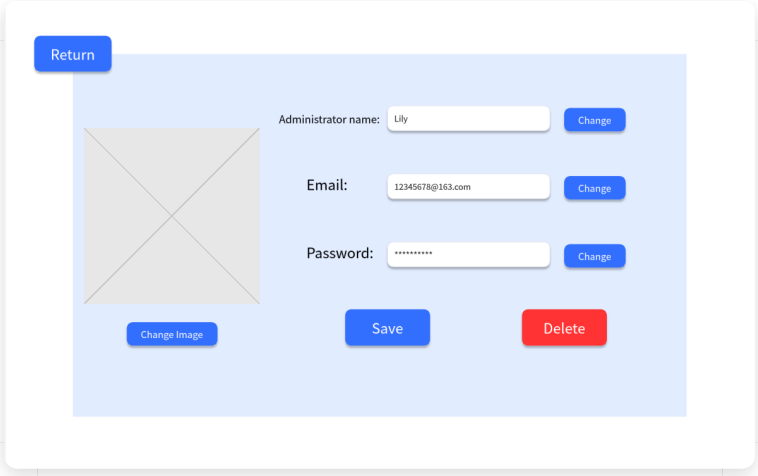
Page Name	Image
Main Admin Page	
Main Organiser Page	
Main User Page	
List Accounts Page	



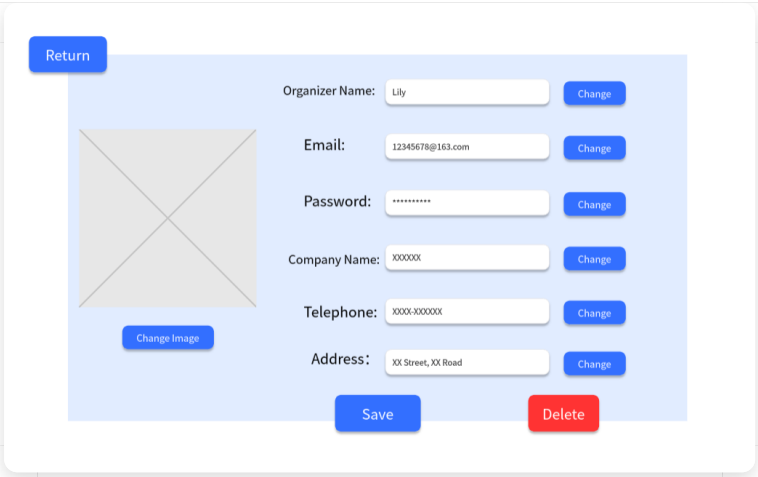
View Account Page (Admin-User)



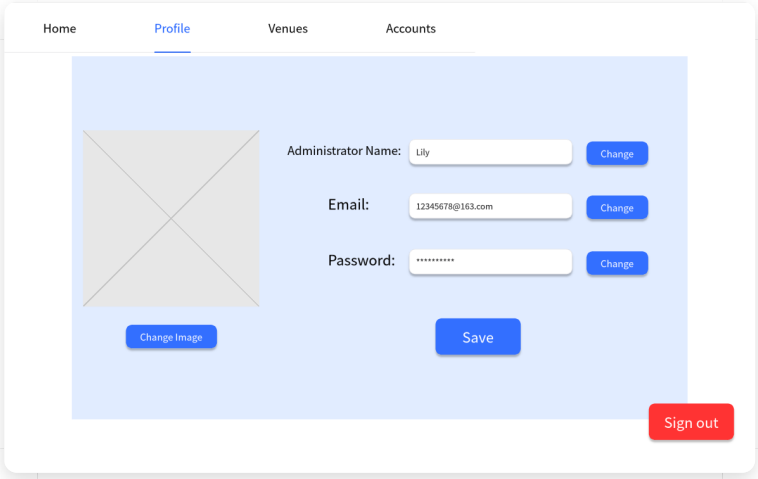
View Account Page (Admin-Admin)



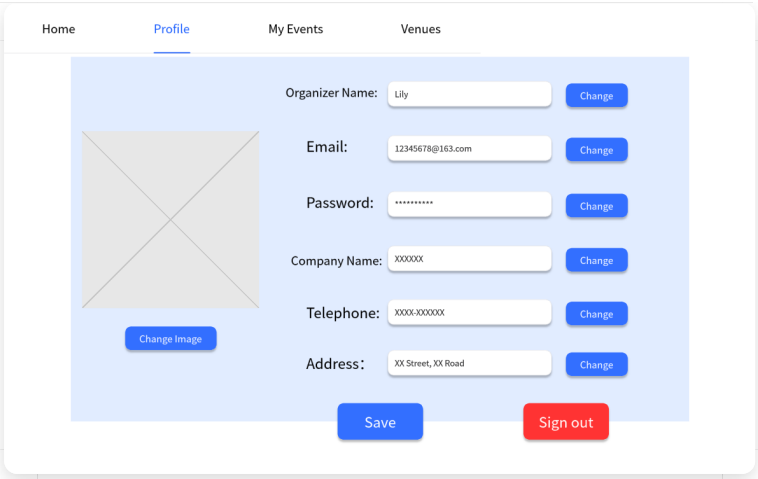
View Account Page (Admin-Organiser)



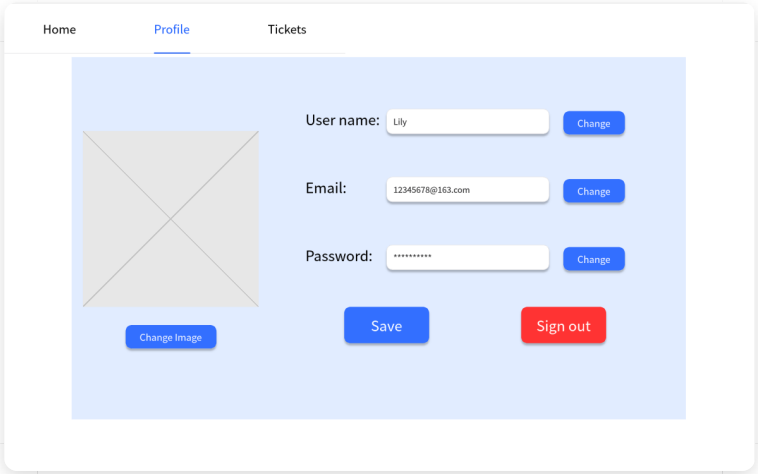
View Account Page (Admin)



View Account Page (Organiser)

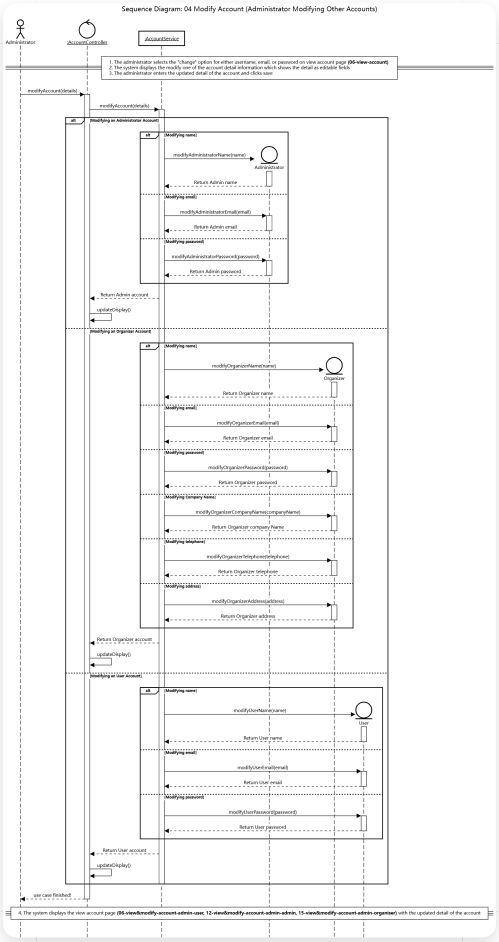


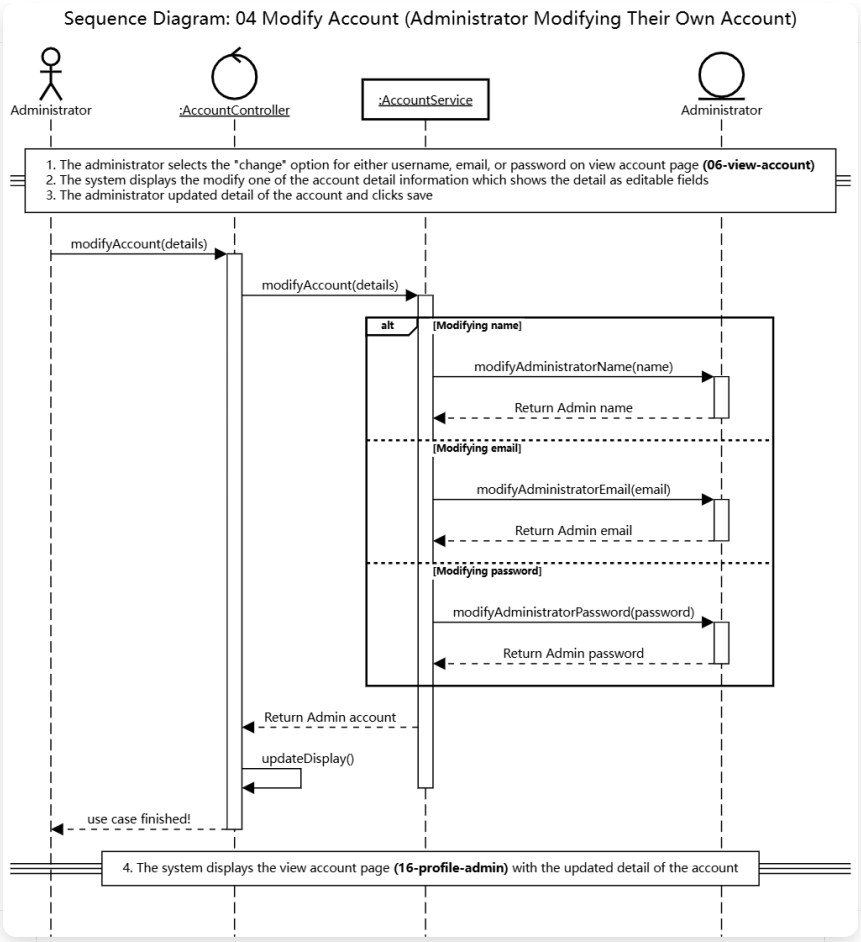
View Account Page (User)

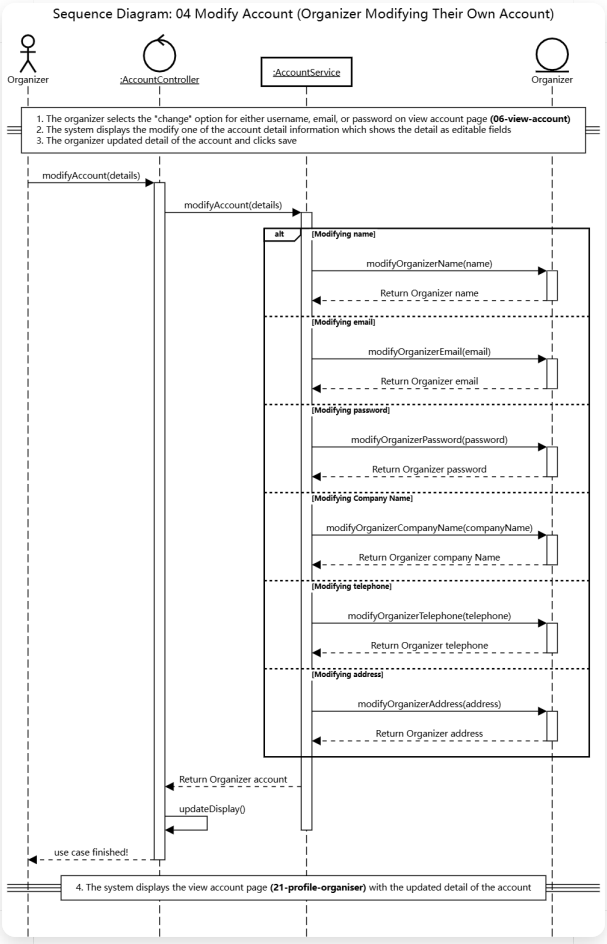


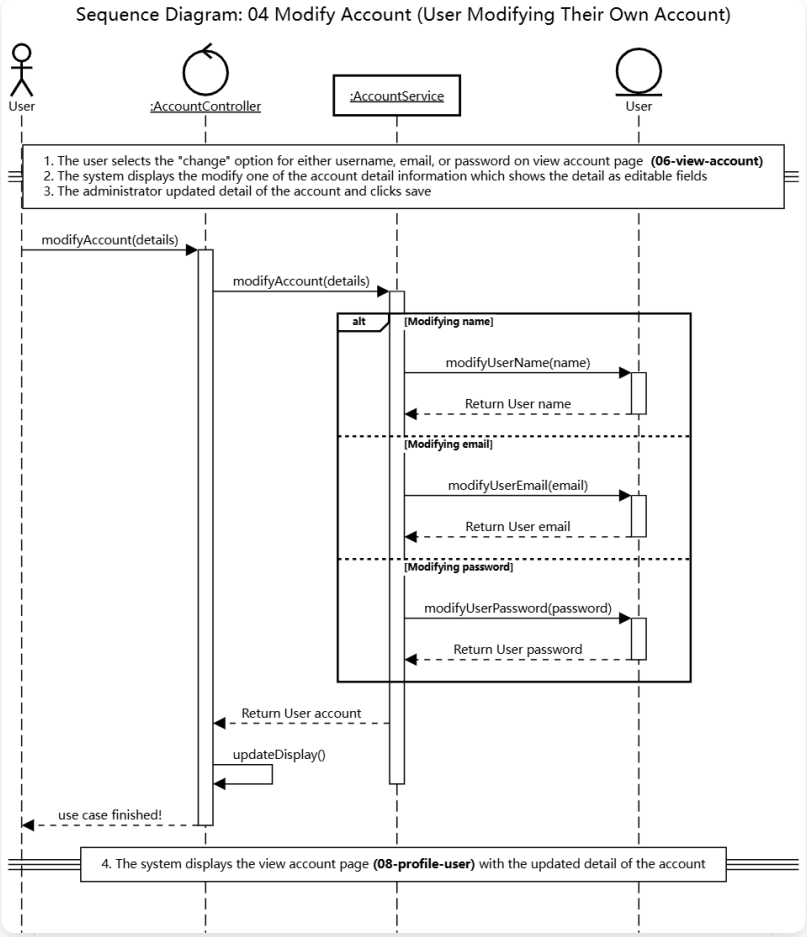
Use Case 04 - Modify Account

01 - Basic Course of Events

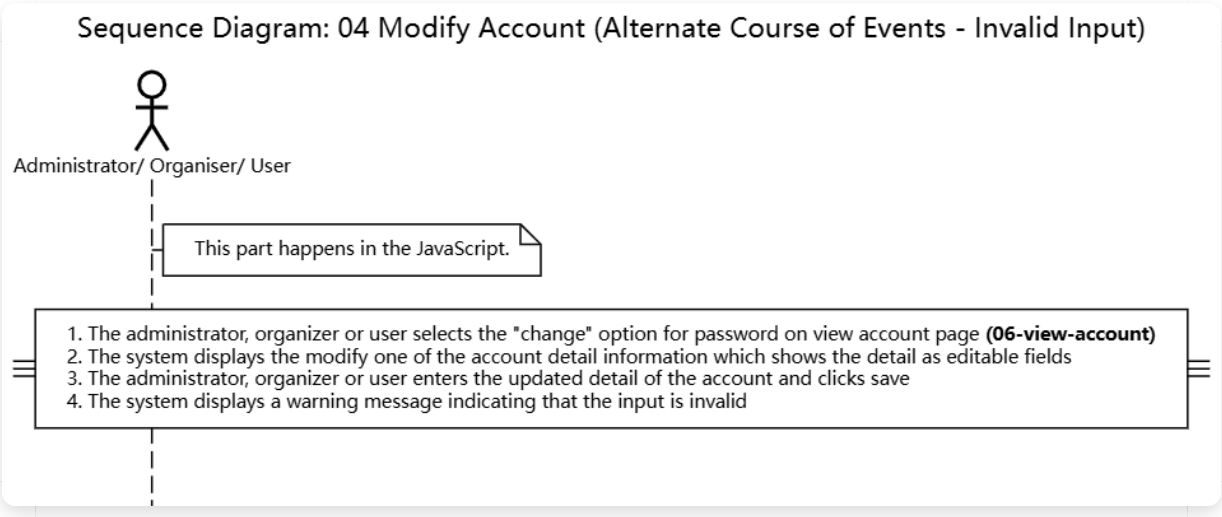



Sequence Diagram: 04 Modify Account (Administrator Modifying Their Own Account)
Administrator
:AccountController
:AccountService
Administrator
1. The administrator selects the "change" option for either username, email, or password on view account page (06-view-account)
2. The system displays the modify one of the account detail information which shows the detail as editable fields
3. The administrator updated detail of the account and clicks save
modifyAccount(details)
modifyAccount(details)
alt
[Modifying name]
modifyAdministratorName(name)
Return Admin name
[Modifying email]
modifyAdministratorEmail(email)
Return Admin email
[Modifying password]
modifyAdministratorPassword(password)
Return Admin password
Return Admin account
updateDisplay()
use case finished!
4. The system displays the view account page (16-profile-admin) with the updated detail of the account


Sequence Diagram: 04 Modify Account (Organizer Modifying Their Own Account)
Organizer
:AccountController
:AccountService
Organizer
1. The organizer selects the "change" option for either username, email, or password on view account page (06-view-account)
2. The system displays the modify one of the account detail information which shows the detail as editable fields
3. The organizer updated detail of the account and clicks save
modifyAccount(details)
modifyAccount(details)
alt
[Modifying name]
modifyOrganizerName(name)
Return Organizer name
[Modifying email]
modifyOrganizerEmail(email)
Return Organizer email
[Modifying password]
modifyOrganizerPassword(password)
Return Organizer password
[Modifying Company Name]
modifyOrganizerCompanyName(companyName)
Return Organizer company Name
[Modifying telephone]
modifyOrganizerTelephone(telephone)
Return Organizer telephone
[Modifying address]
modifyOrganizerAddress(address)
Return Organizer address
Return Organizer account
updateDisplay()
use case finished!
4. The system displays the view account page (21-profile-organiser) with the updated detail of the account



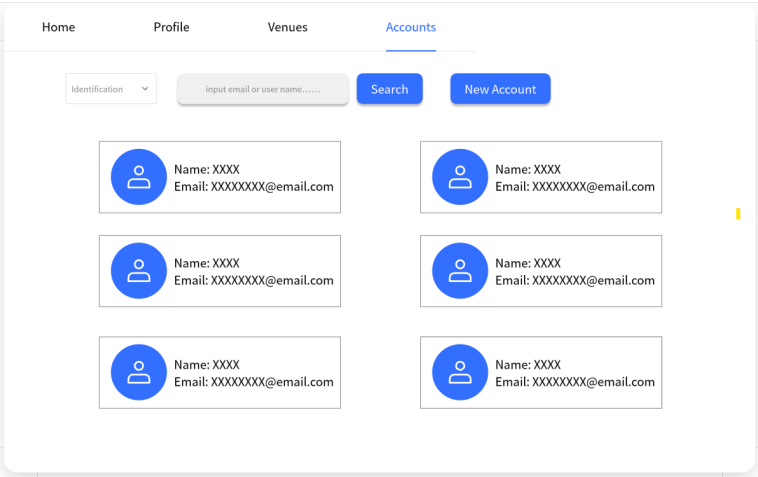
02 - Alternate Course of Events



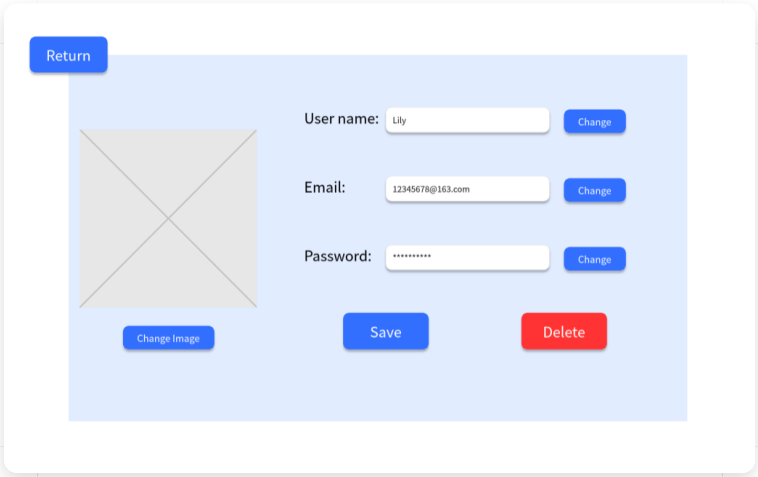
Modify Account - Alternative Courses of Events - Invalid Input

Relevant UI Sketches

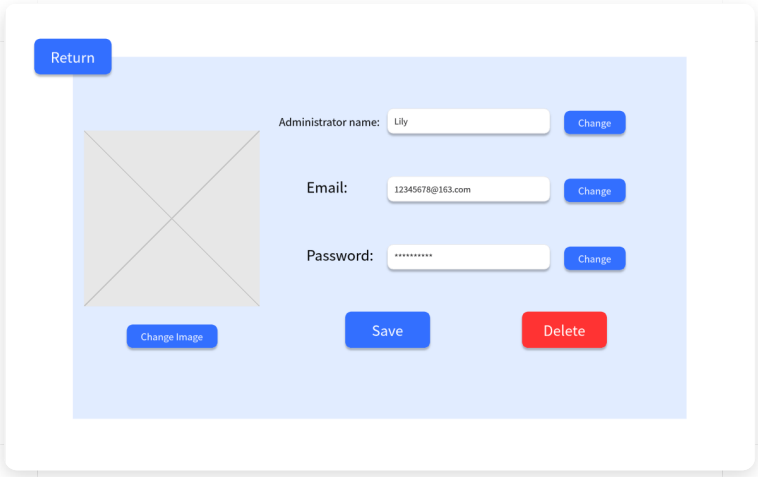
Page Name	Image
List Accounts Page	



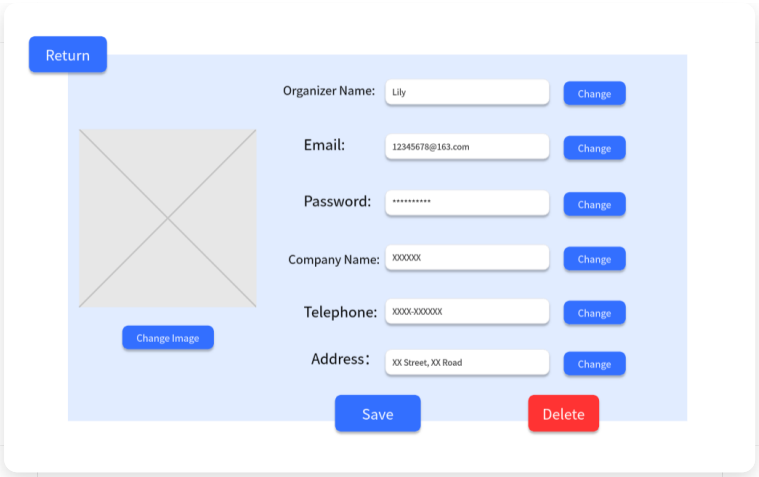
View Account Page (Admin-User)



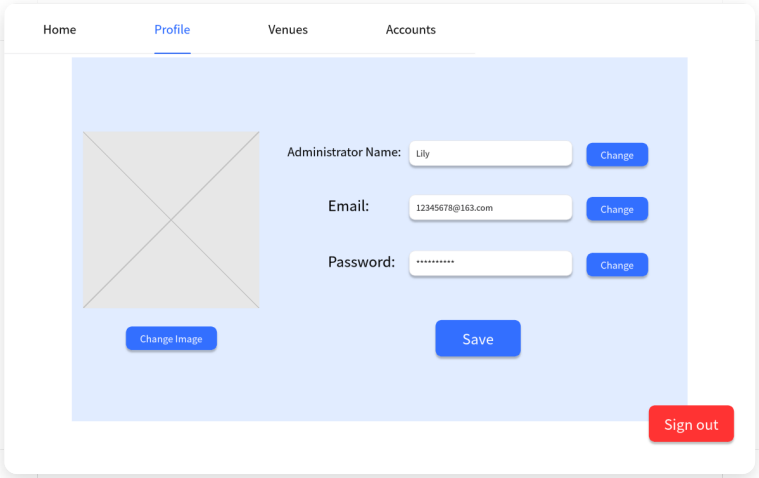
View Account Page (Admin-Admin)



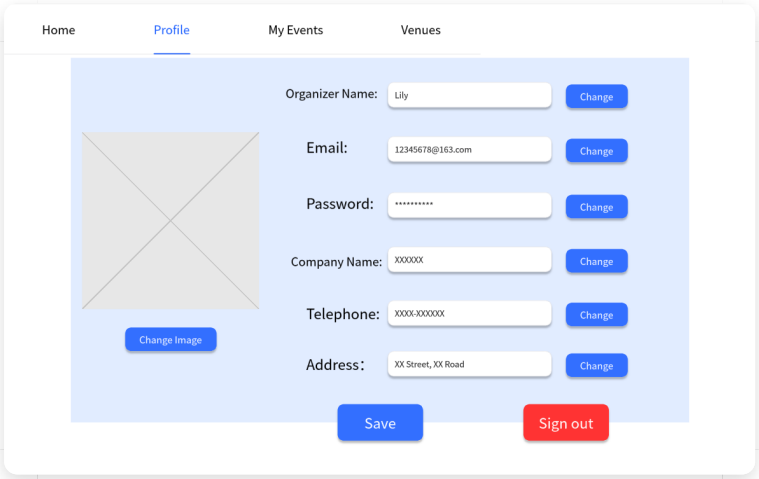
View Account Page (Admin-Organiser)



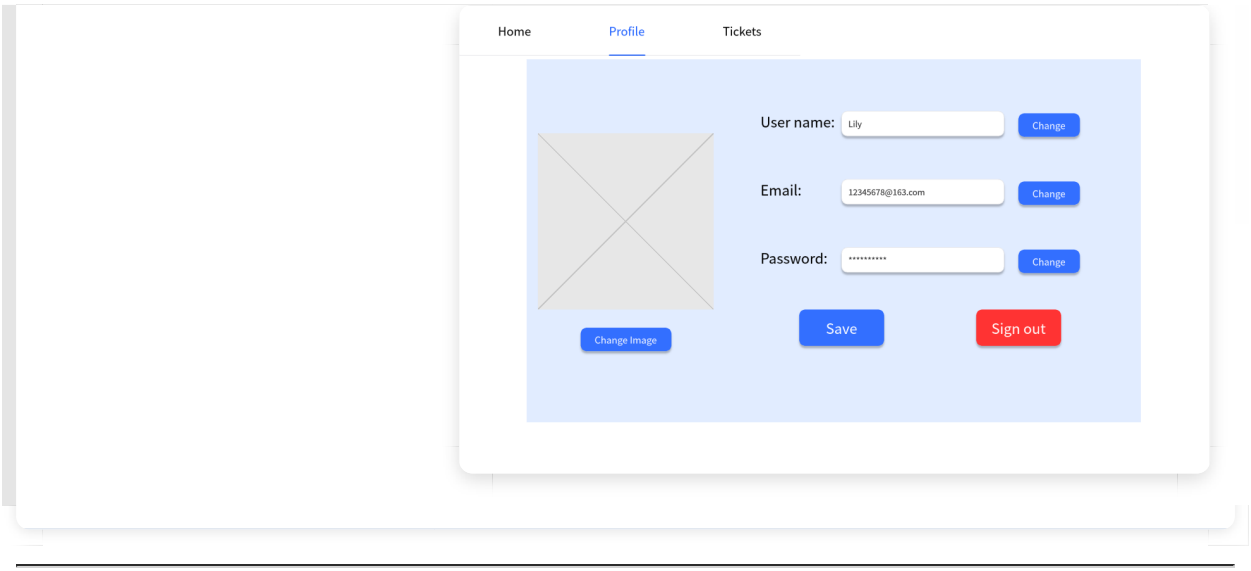
View Account Page (Admin)



View Account Page (Organiser)

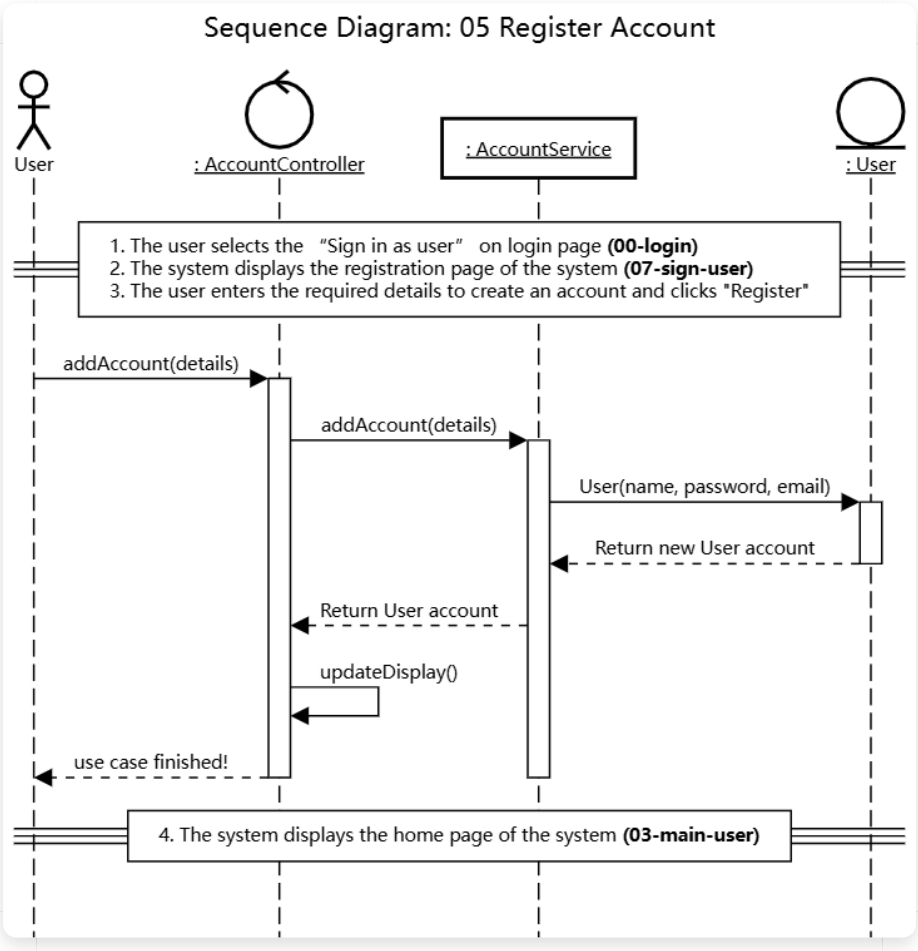


View Account Page (User)


Home
Profile
Tickets
User name:
Lily
Change
Email:
12345678@163.com
Change
Password:
Change
Change Image
Save
Sign out

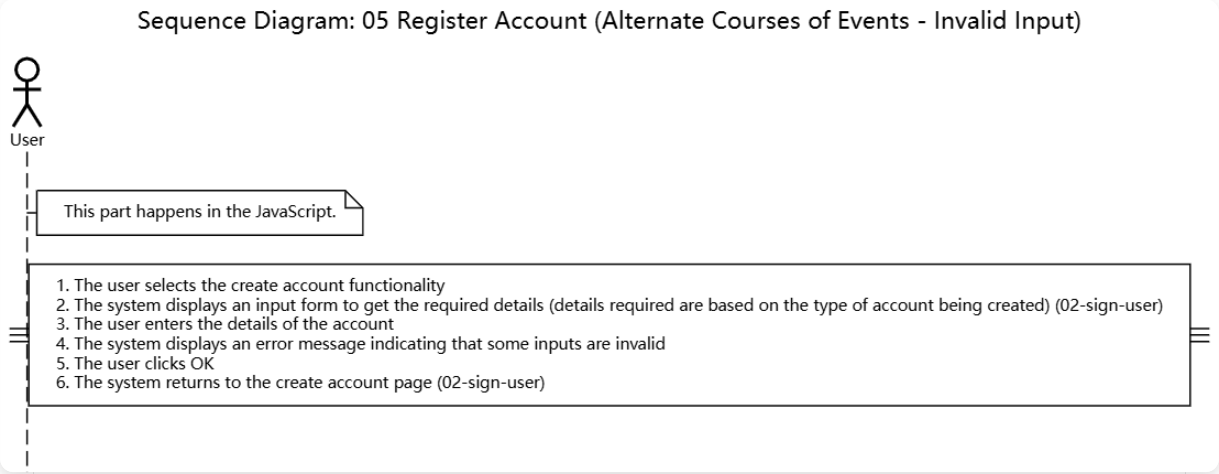
Use Case 05 - Register Account

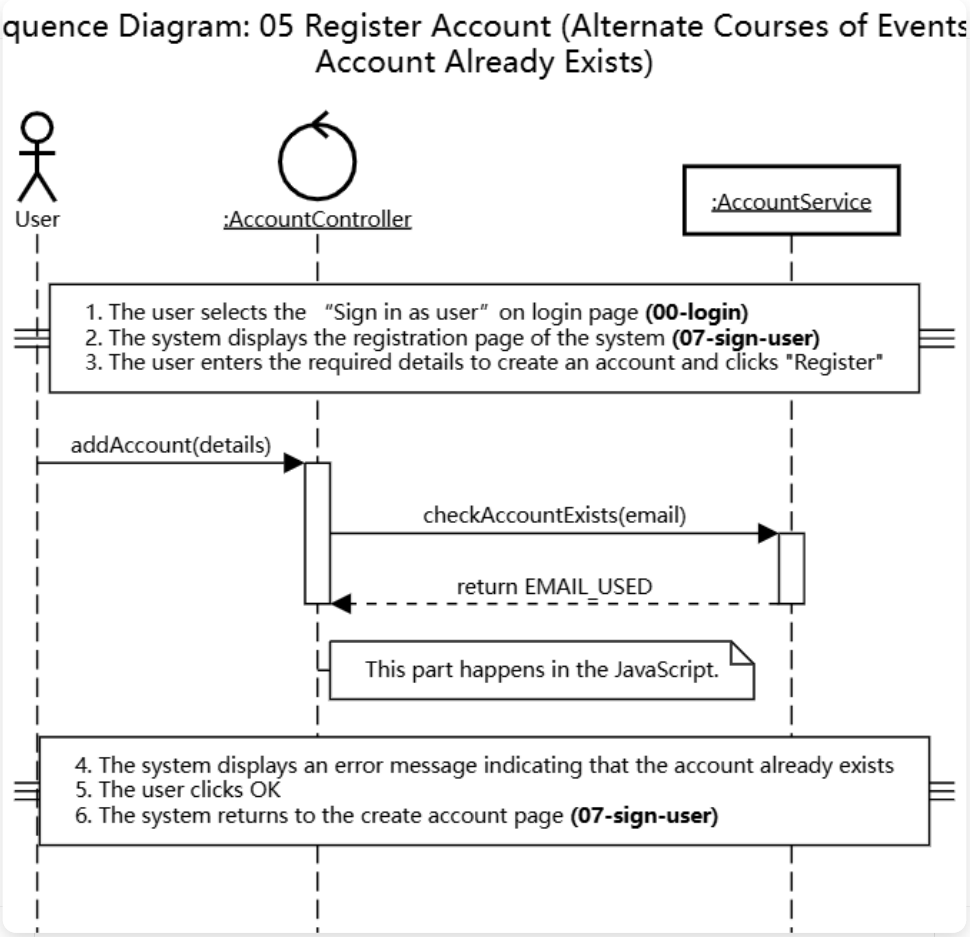
01 - Basic Course of Events



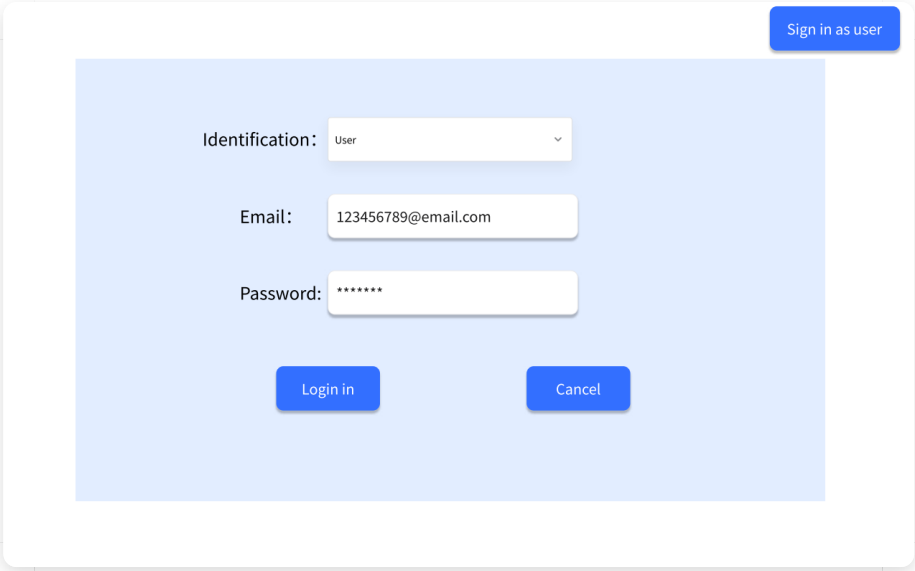
Register Account - Basic Course of Events(Administrator Modifying Other Accounts)

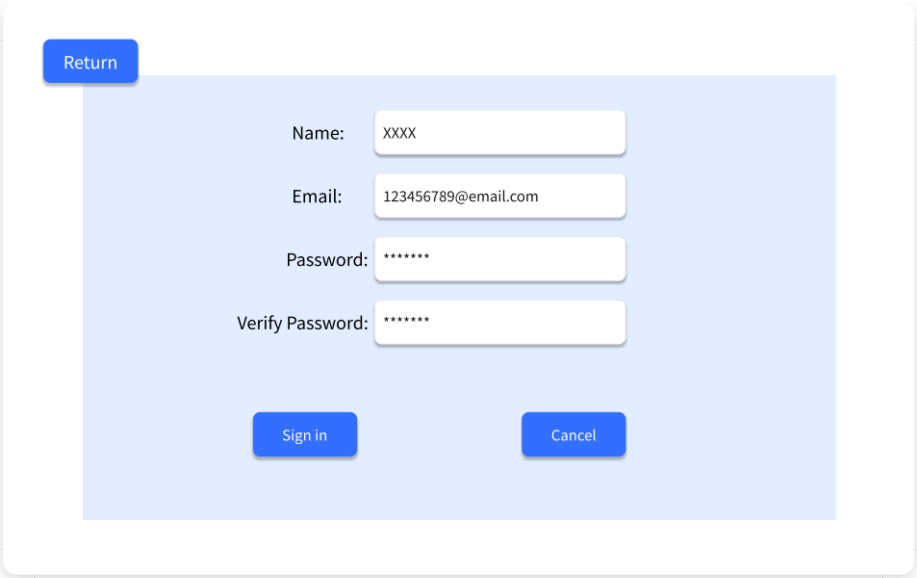
02 - Alternate Course of Events



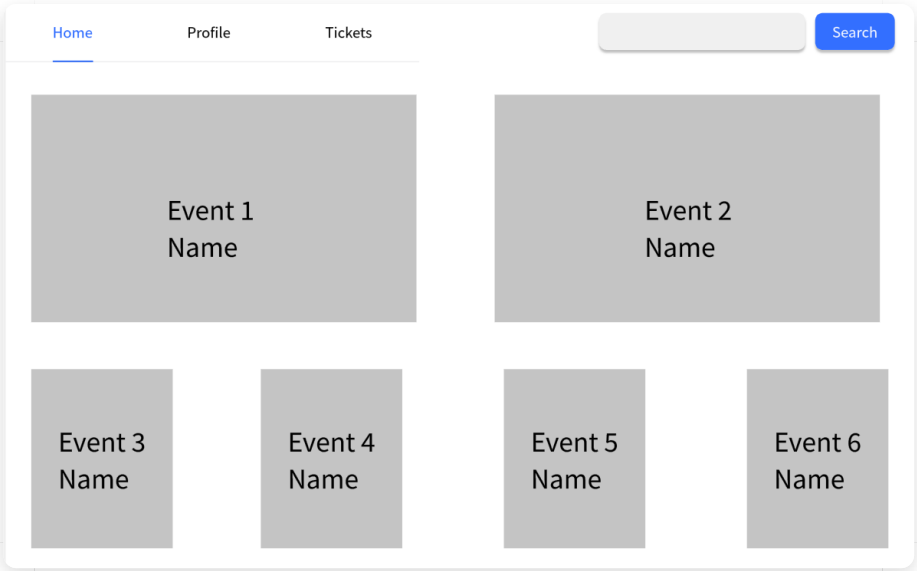


Relevant UI Sketches

Page Name	Image
Login Page	
Sign Page	



Main User Page

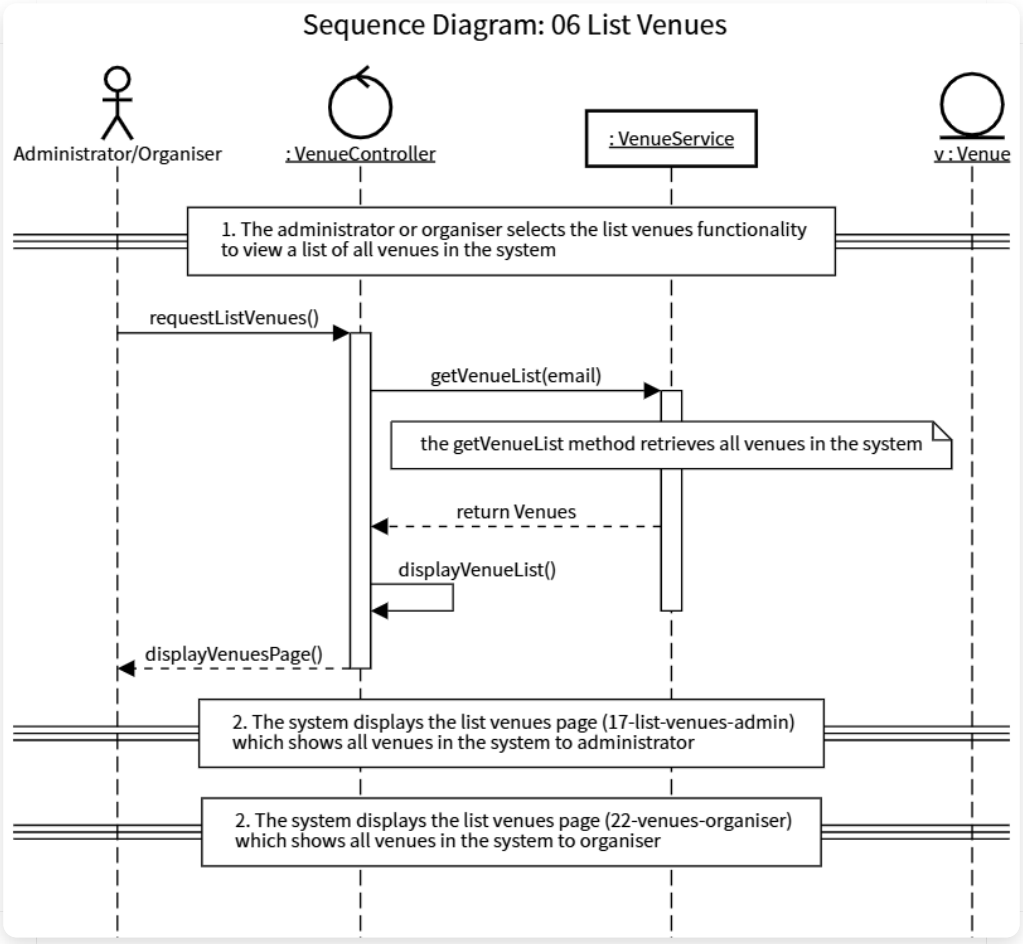


Duplicate Email Message

Duplicate Email Message

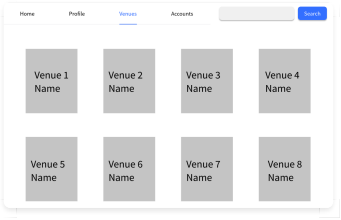
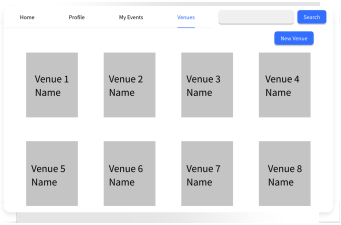
Use Case 06 - List Venues

01 - Basic Course of Events



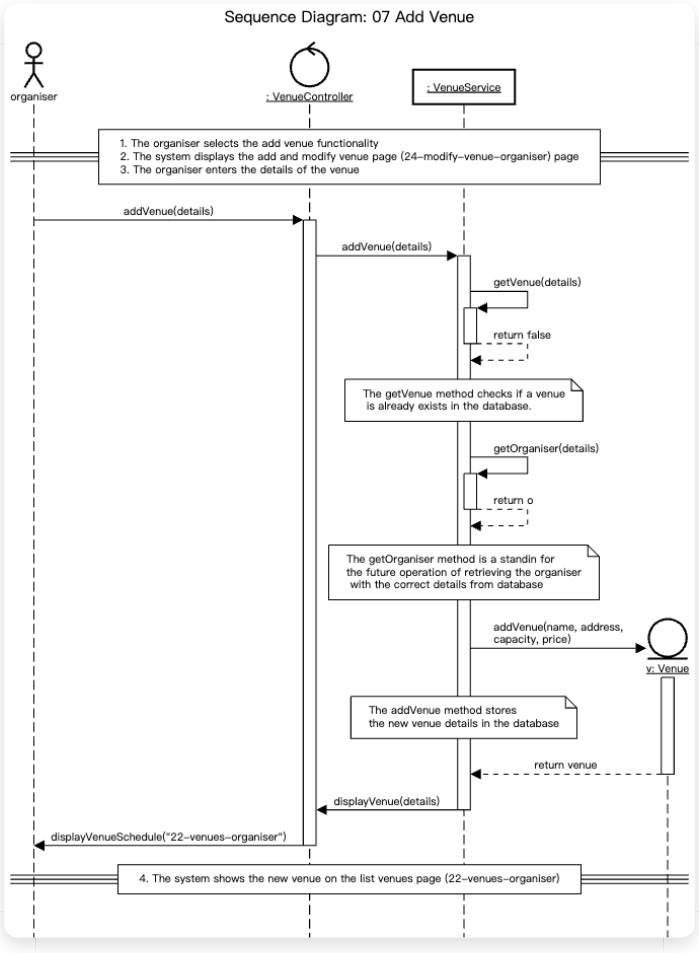
List Venues - Basic Course of Events

Related UI Prototypes

Page Name	Image
17-list-venues-admin	
22-venues-organiser	

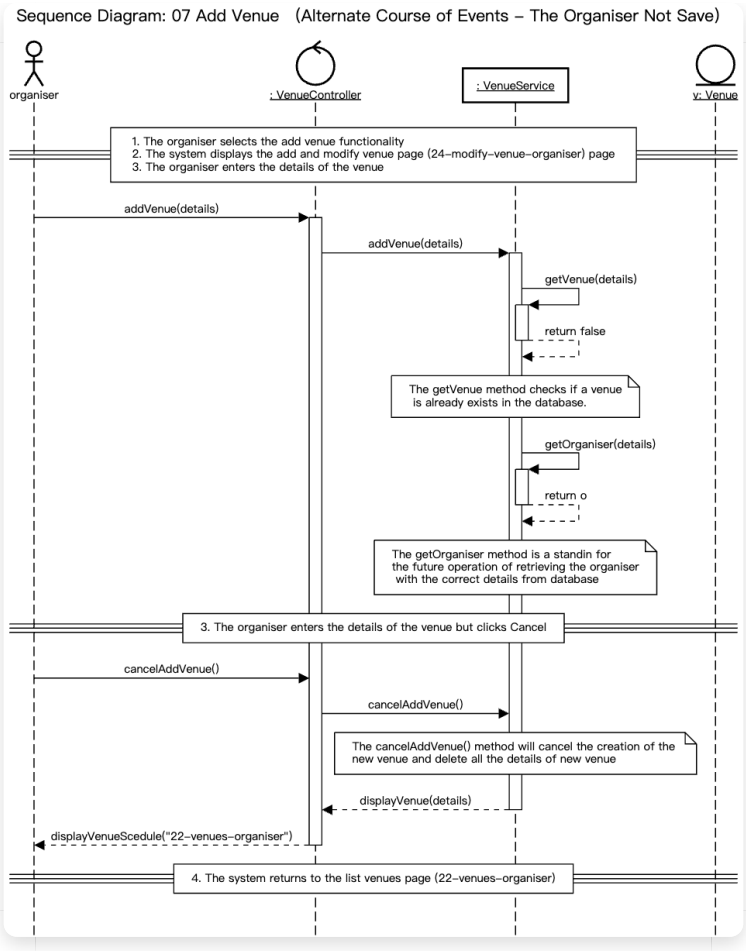
Use Case 07 - Add Venue

01 - Basic Course of Events



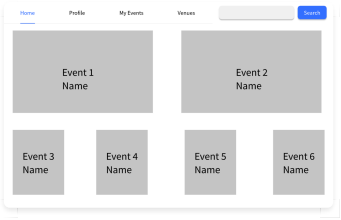
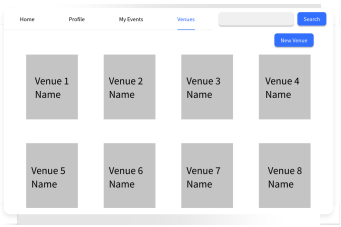
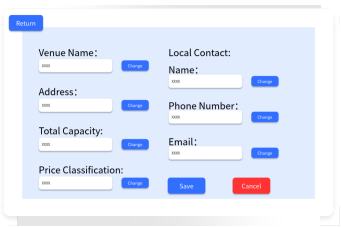
Add Venue - Basic Course of Events

02 - Alternate Course of Events - The Organiser Not Save



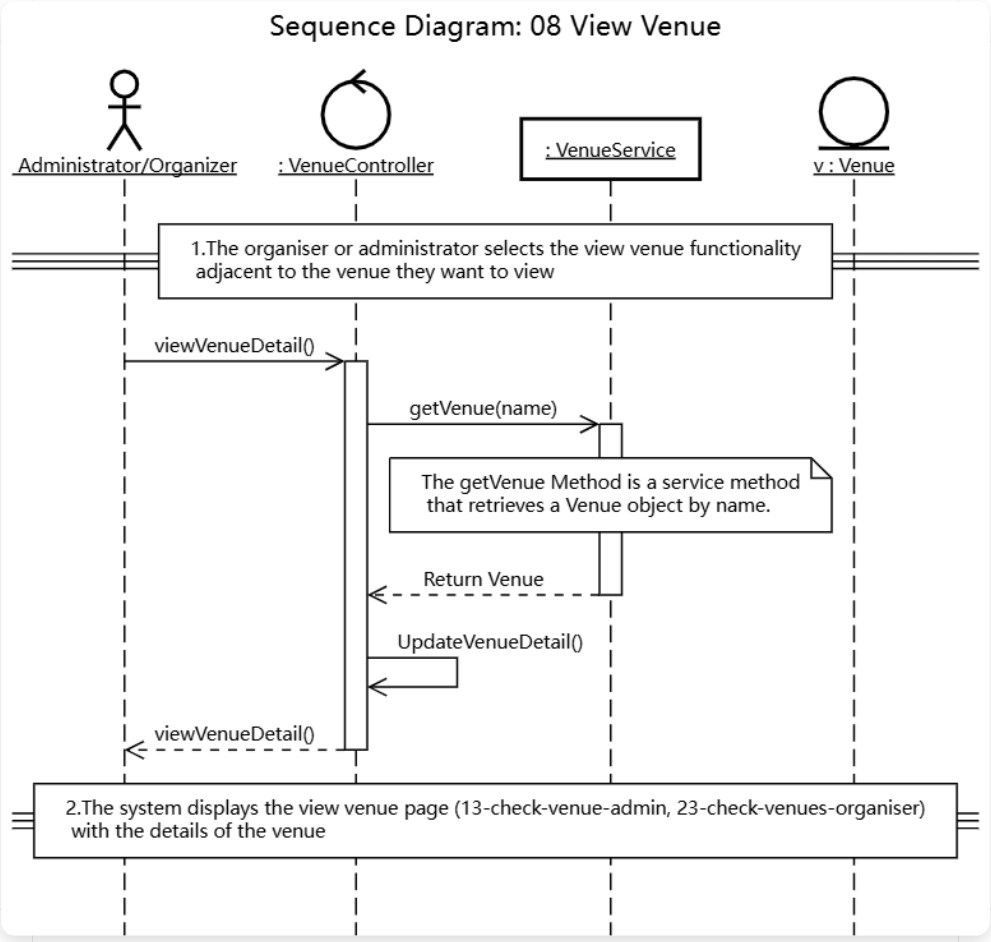
Add Venue - Alternate Course of Events - The Organiser Not Save

Related UI Prototypes

Page Name	Image
Organiser Main Page	
The List Venues Page(Organiser)	
The Add and Modify Venue Page	

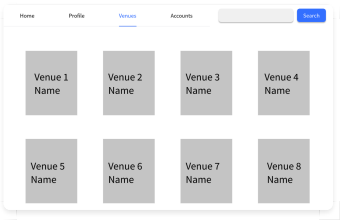
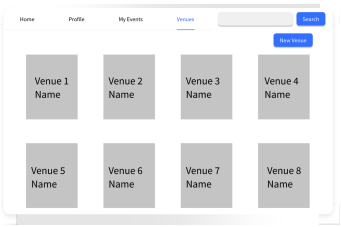
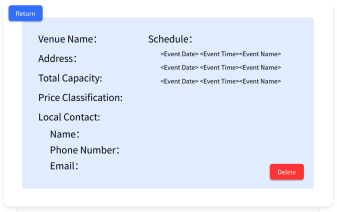
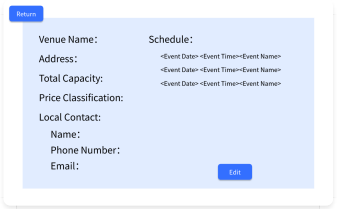
Use Case 08 - View Venue

01 - Basic Course of Events



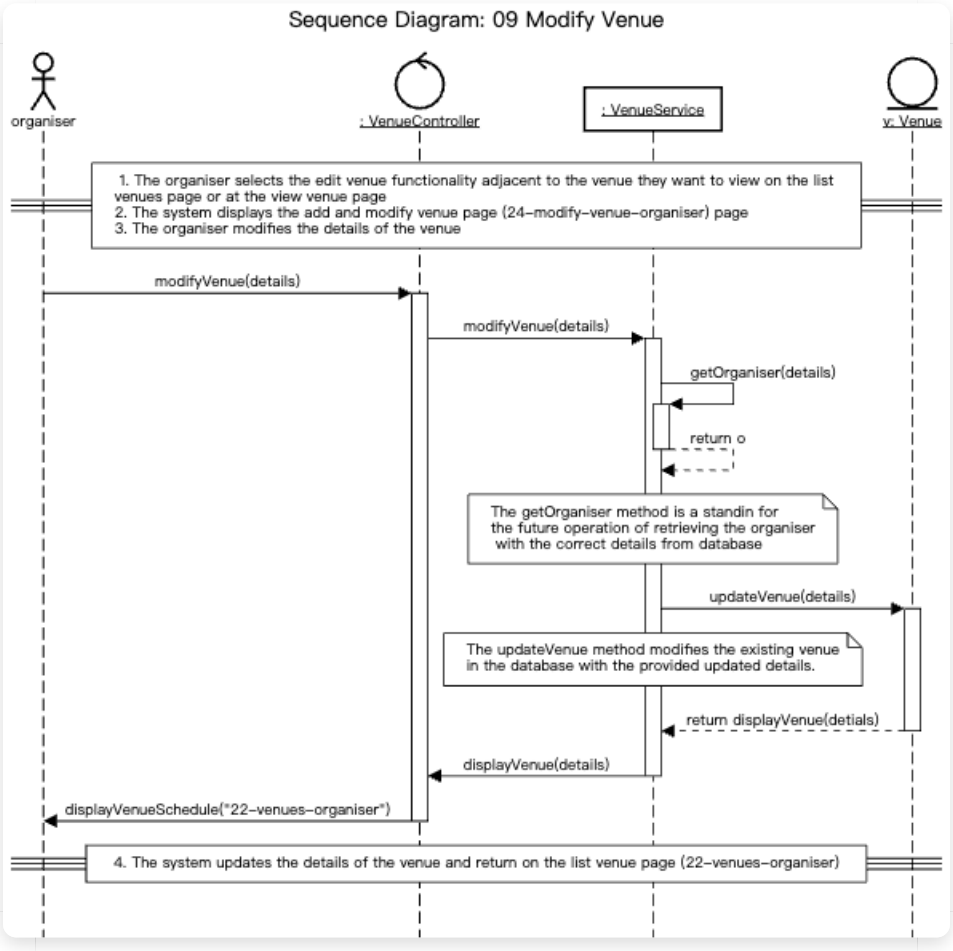
LIst Account - Basic Course of Events

Related UI Prototypes

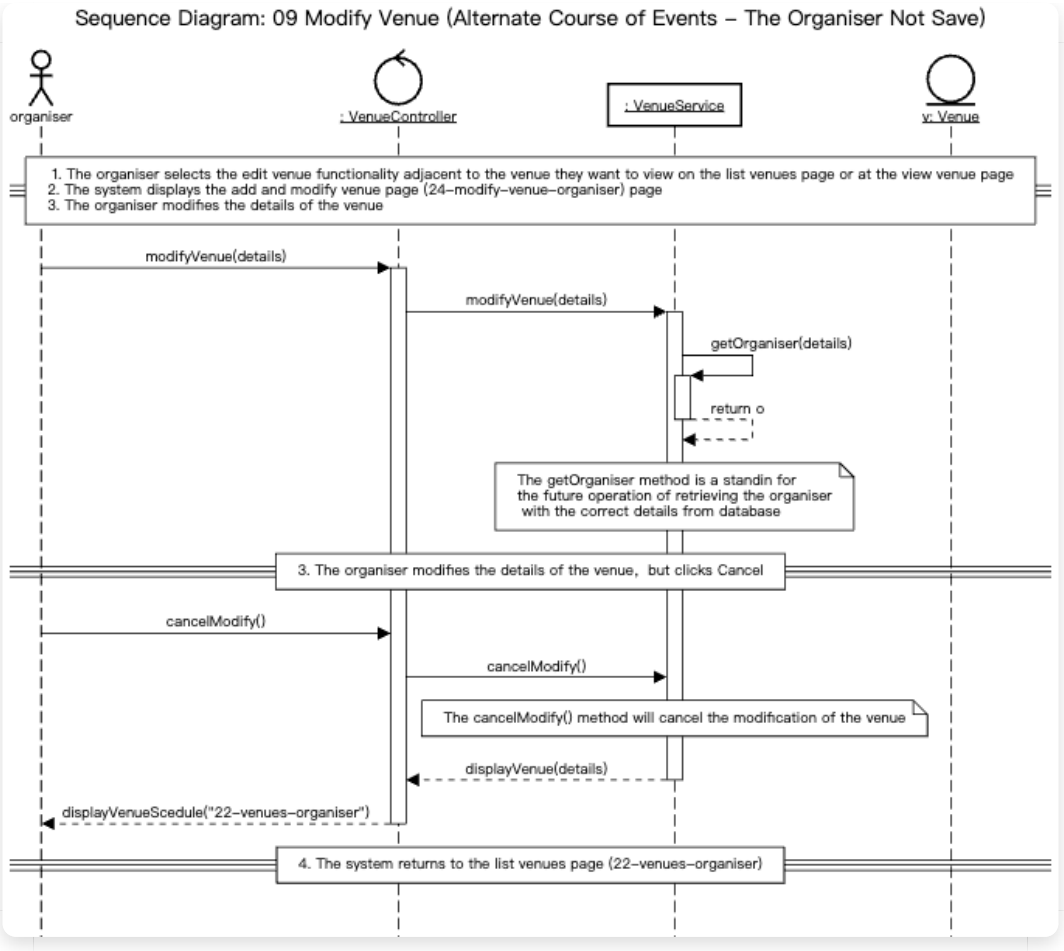
Page Name	Image
List Venues Page (Admin)	
List Venues Page (Organiser)	
View Venue Page (Admin)	
View Venue Page (Organiser)	

Use Case 09 - Modify Venue

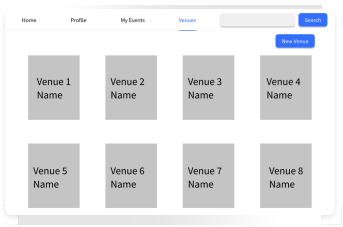
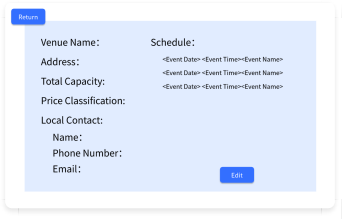
01 - Basic Course of Events



02 - Alternate Course of Events - The Organiser Not Save

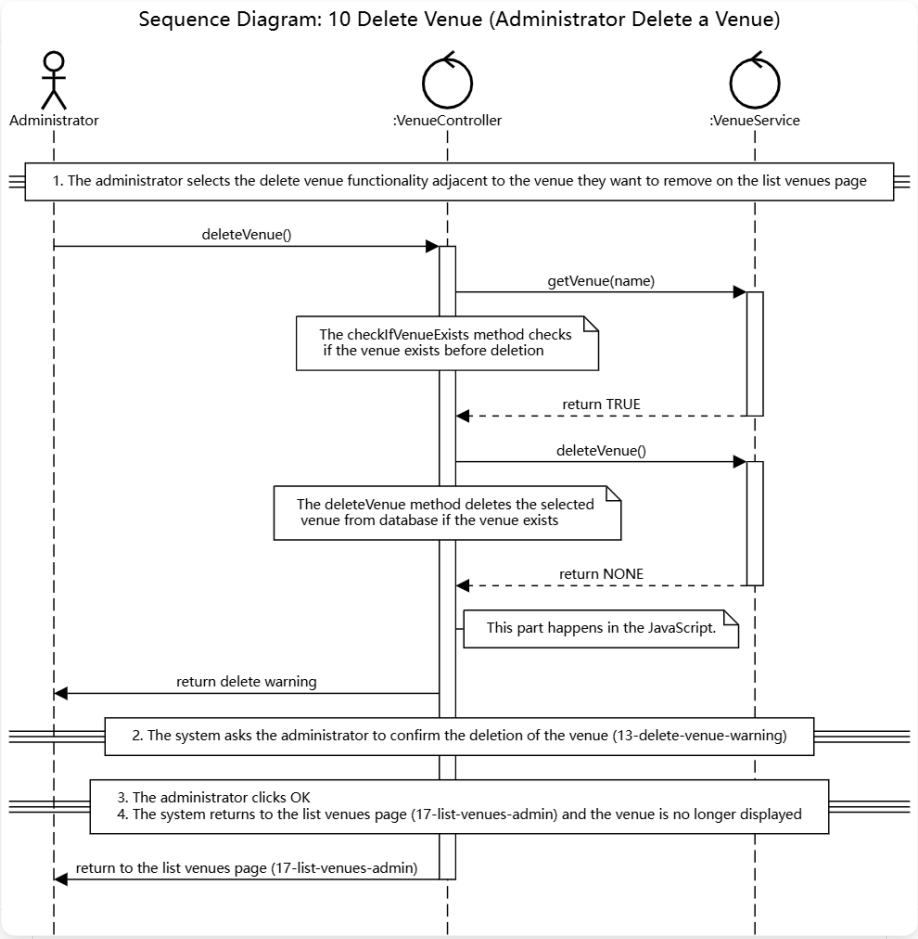


Related UI Prototypes

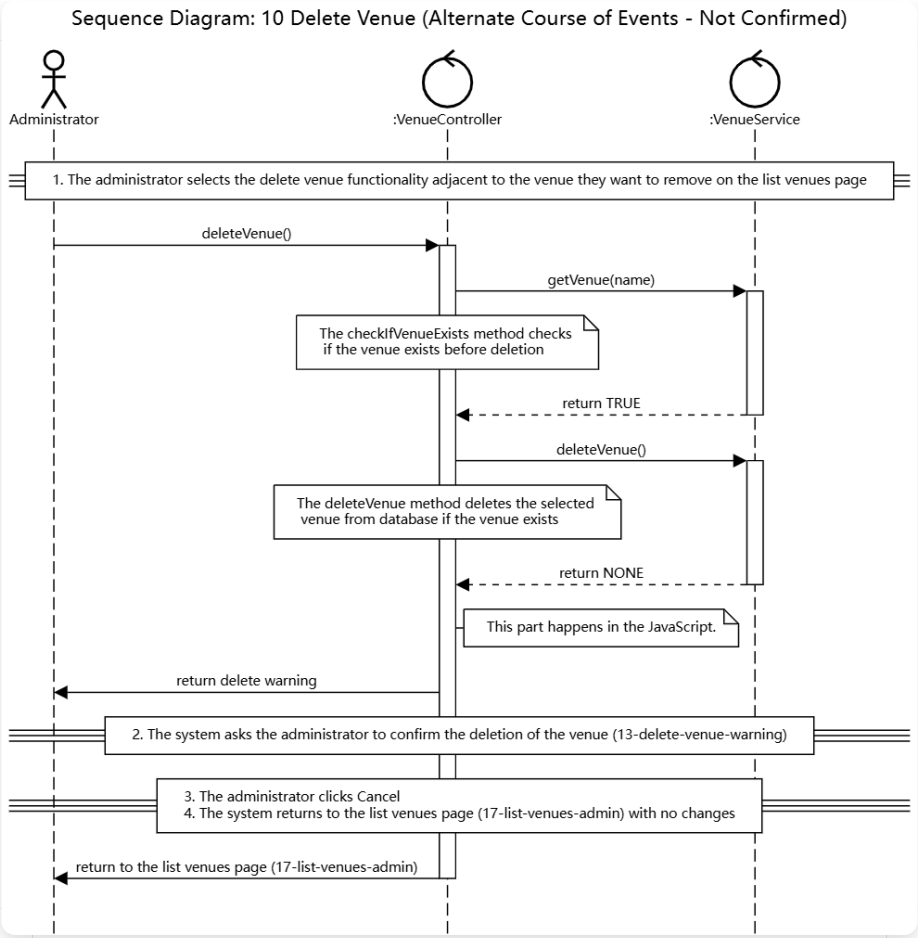
Page Name	Image
The List Venues Page(Organiser)	
The View Venue Page(Organiser)	
The Add and Modify Venue Page	

Use Case 10 - Delete Venue

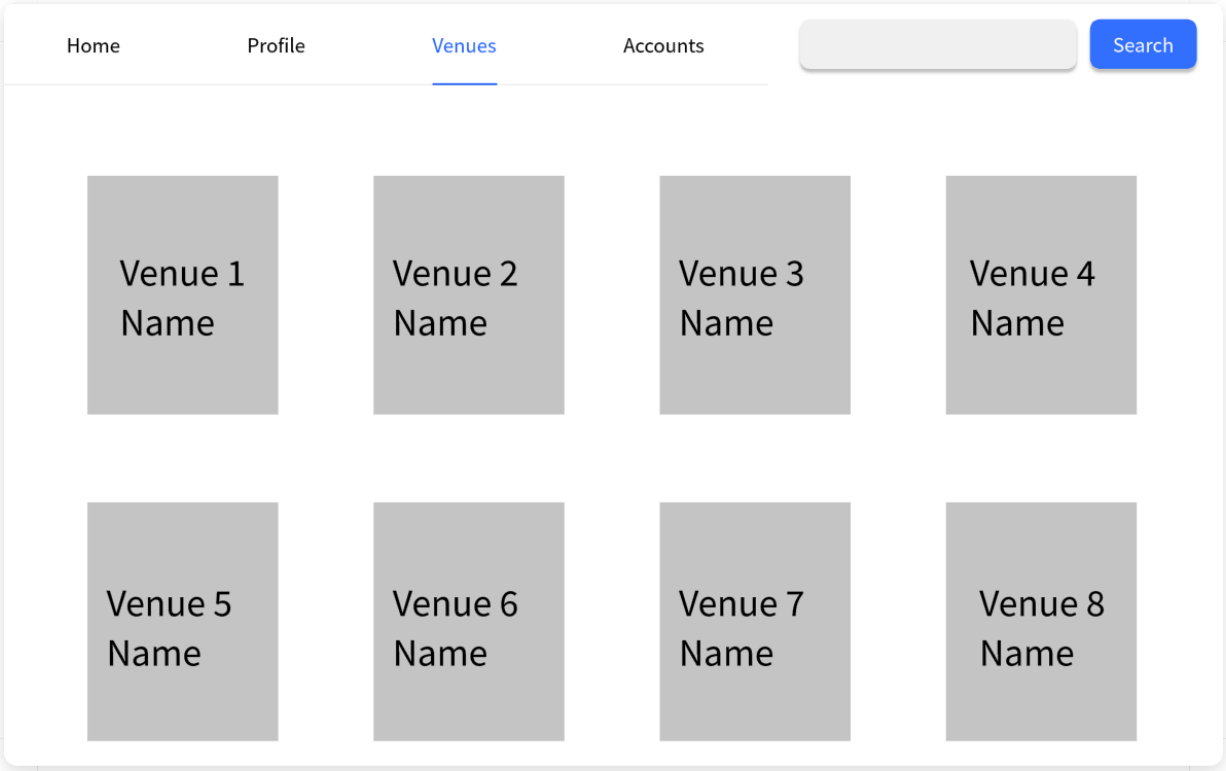
01 - Basic Course of Events



02 - Alternate Course of Events - Not Confirmed



Relevant UI Sketches | Page Name | Image | |—-|——| | List Venues Page (Admin) |

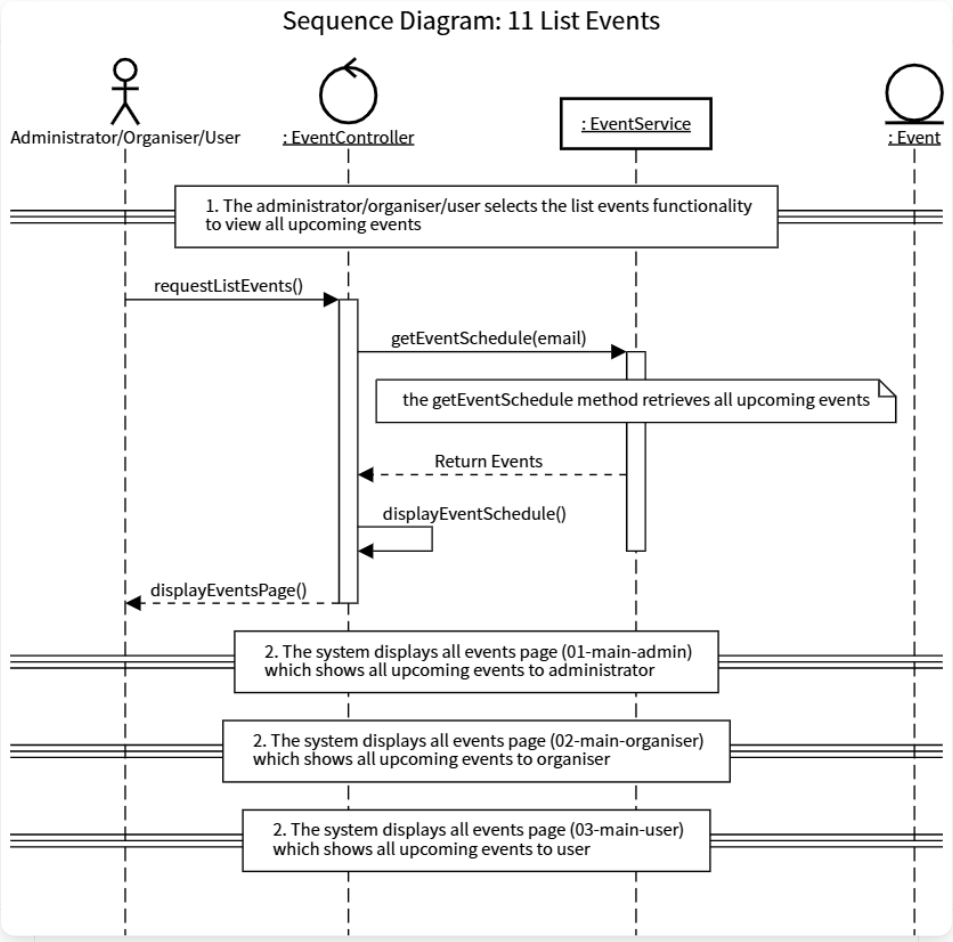


| | Delete Venue Warning |

Delete Venue Warning

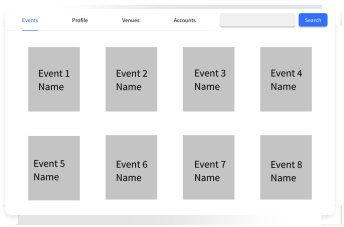
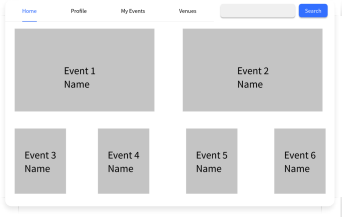
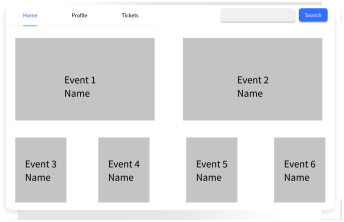
Use Case 11 - List Events

01 - Basic Course of Events



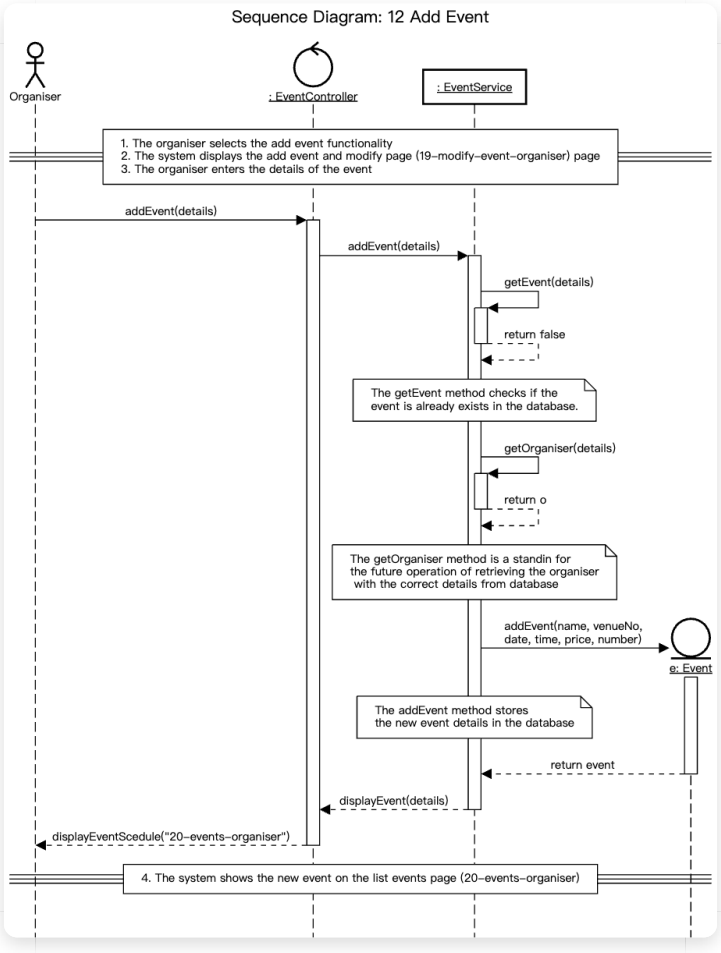
List Eventse - Basic Course of Events

Related UI Prototypes

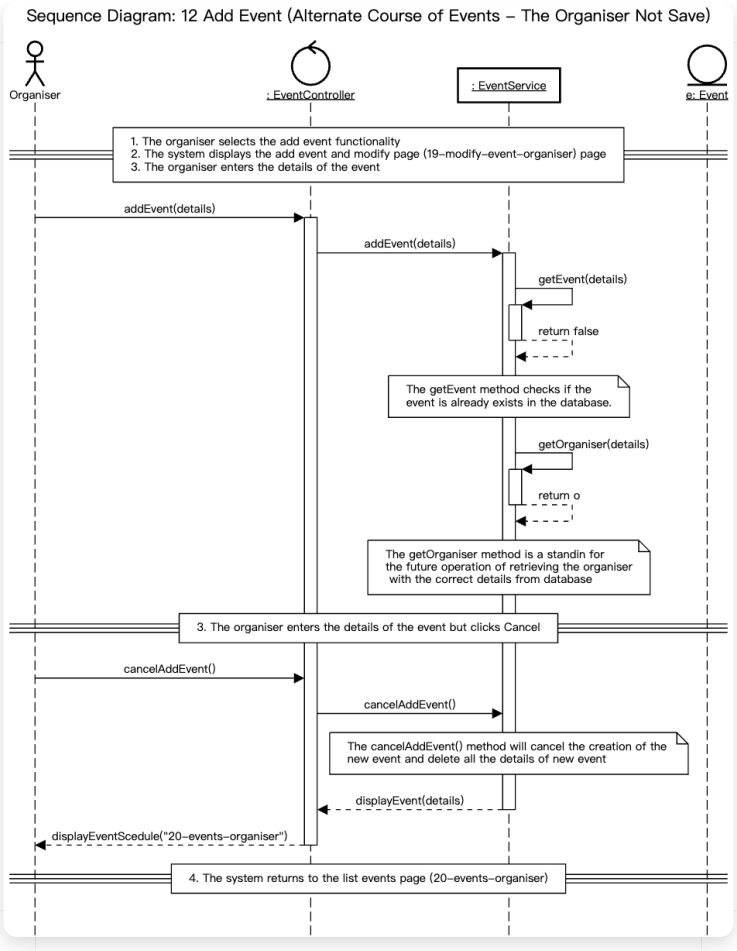
Page Name	Image
01-main-admin	
02-main-organiser	
03-main-user	

Use Case 12 - Add Event

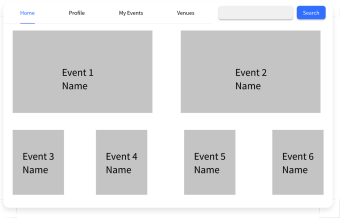
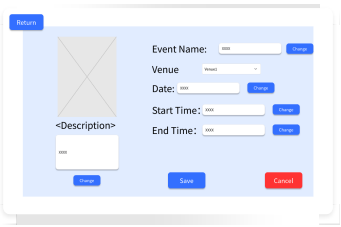
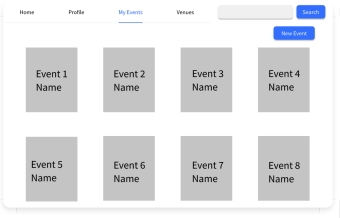
01 - Basic Course of Events



02 - Alternate Course of Events - The Organiser Not Save

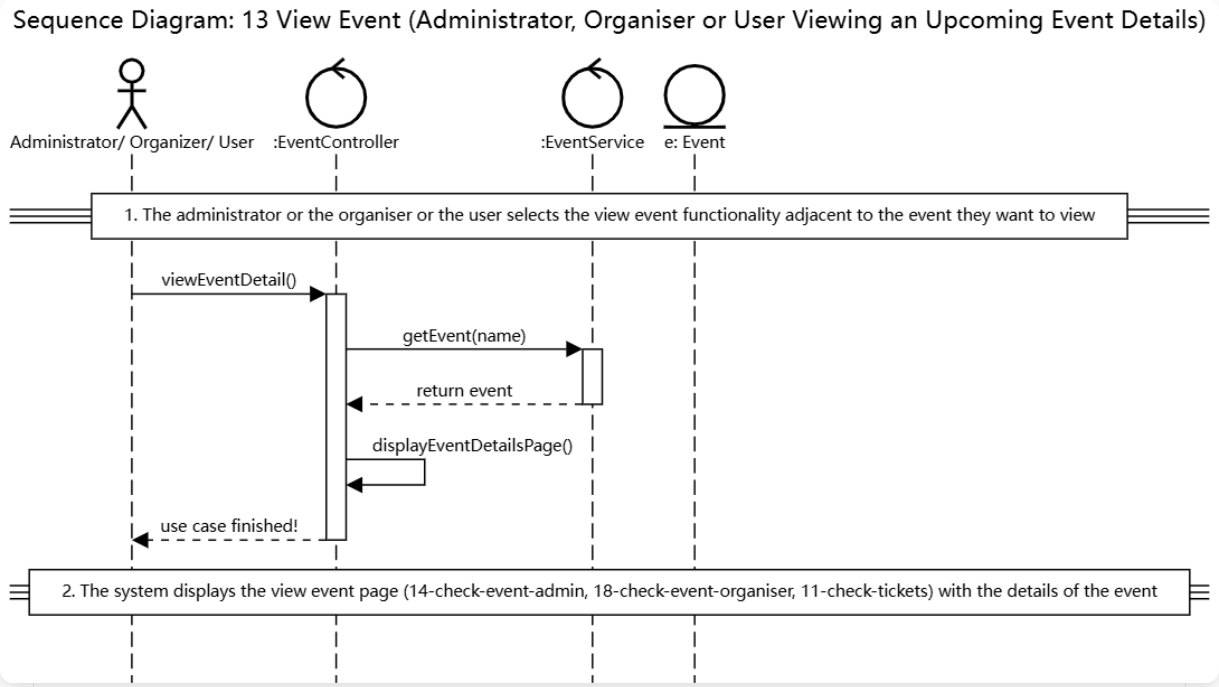


Related UI Prototypes

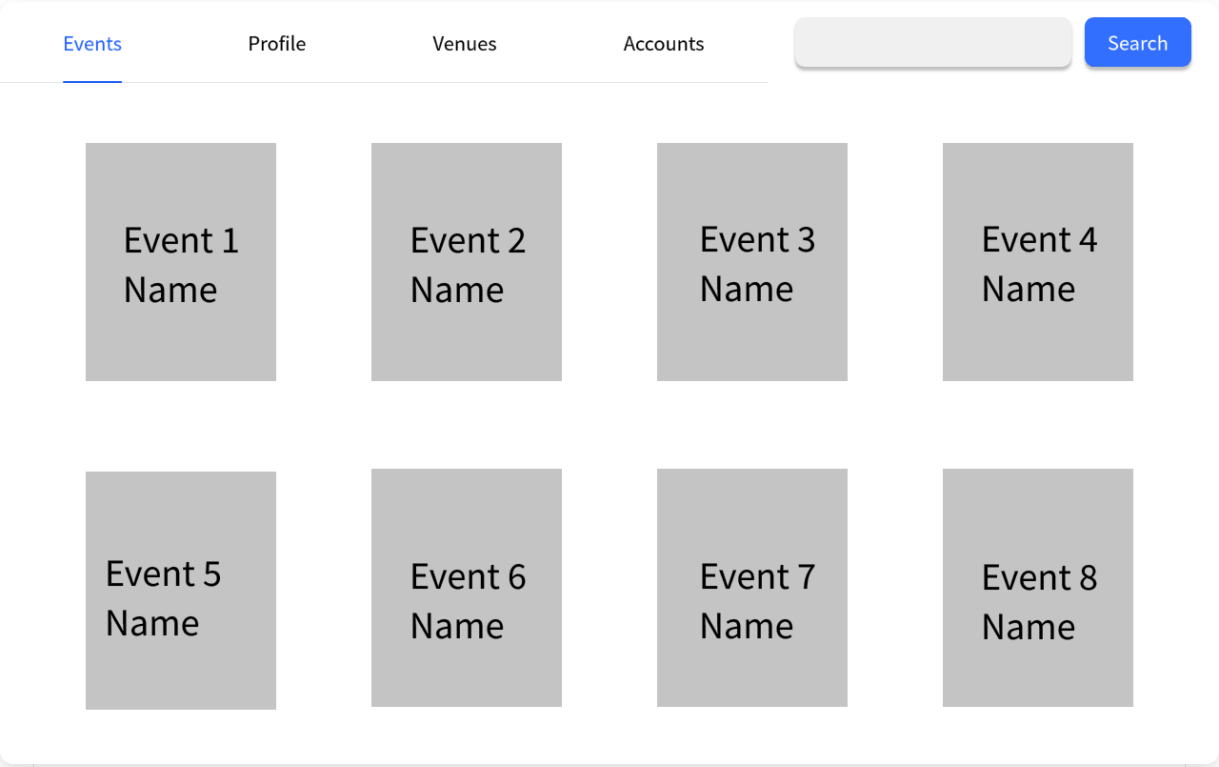
Page Name	Image
Organiser-Main-Page	
The Add and Modify Event Page	
The List Events Page(Organiser)	

Use Case 13 - View Event

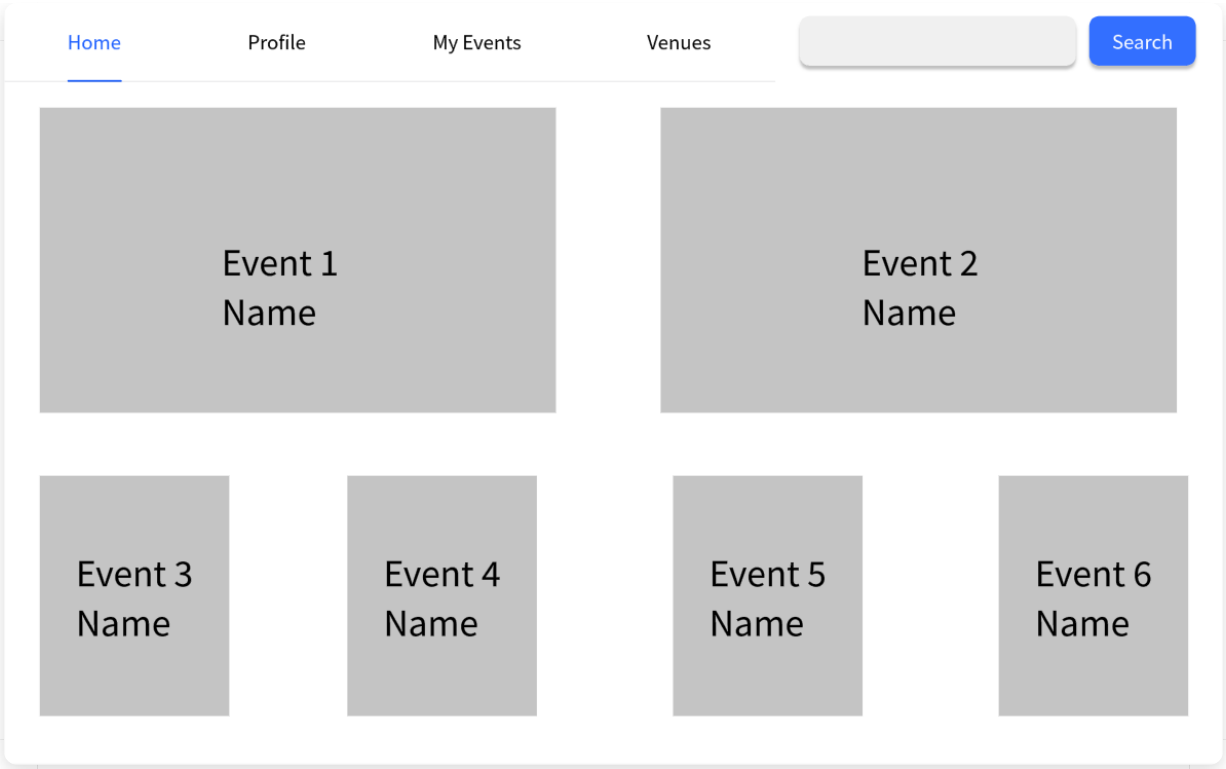
01 - Basic Course of Events



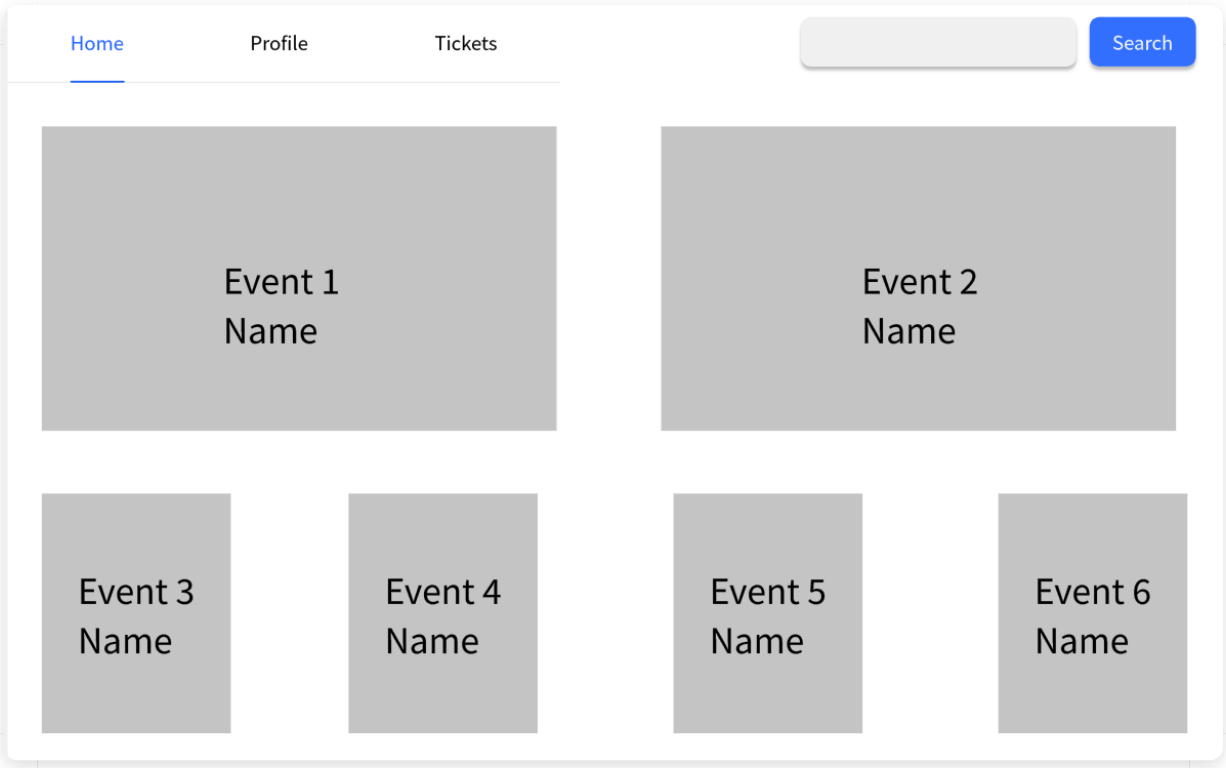
Relevant UI Sketches | Page Name | Image | |———————|———————————————————————————| | Admin Main Page |



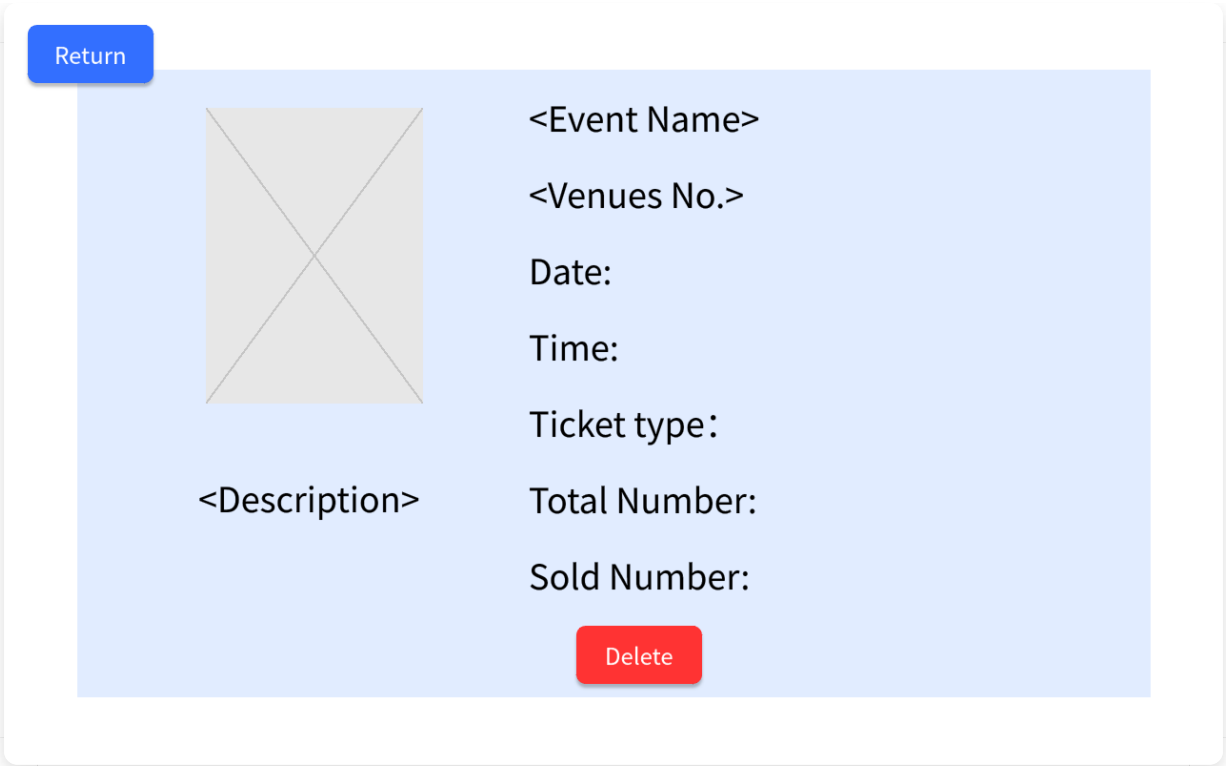
| | Organiser Main Page |



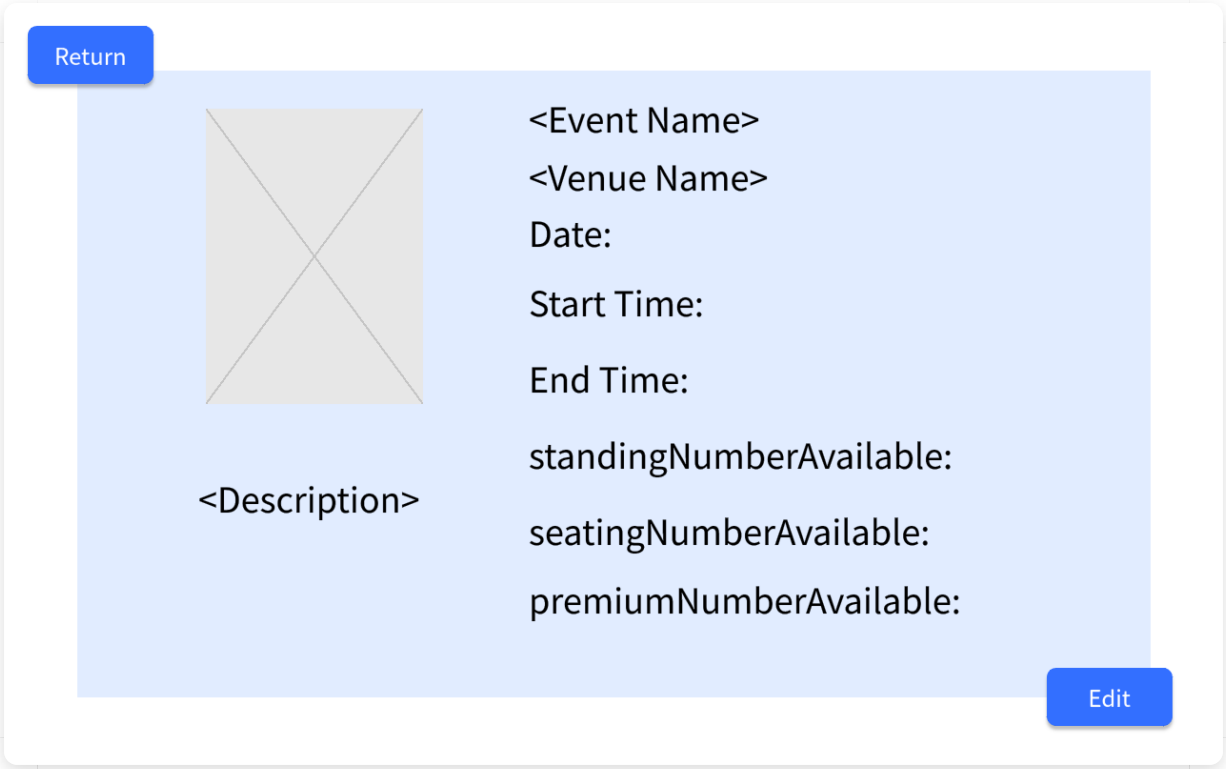
| | User Main Page |



| | View Event Page (Admin) |

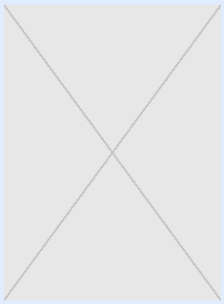


| | View Event Page (Organiser) |



| | View Event Page (User) |

Return



<Description>

<Event Name>

<Venues No.>

<Organizer>

Date:

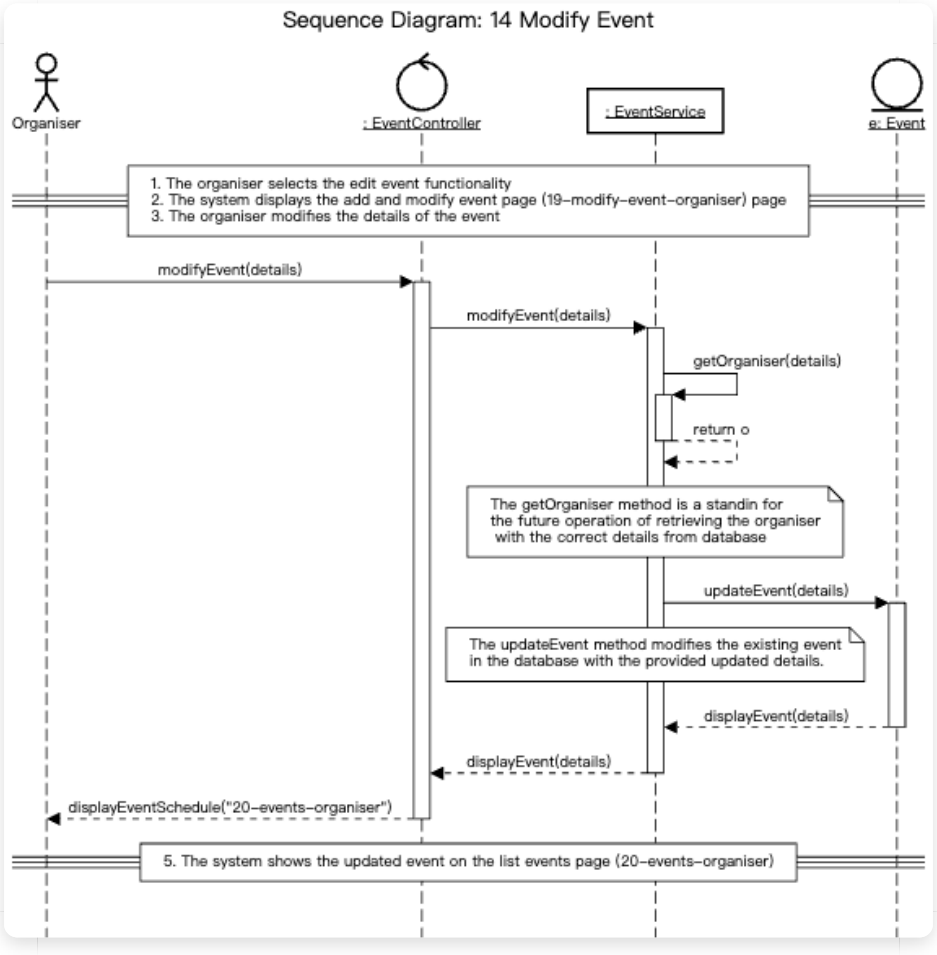
Time:

Ticket type:

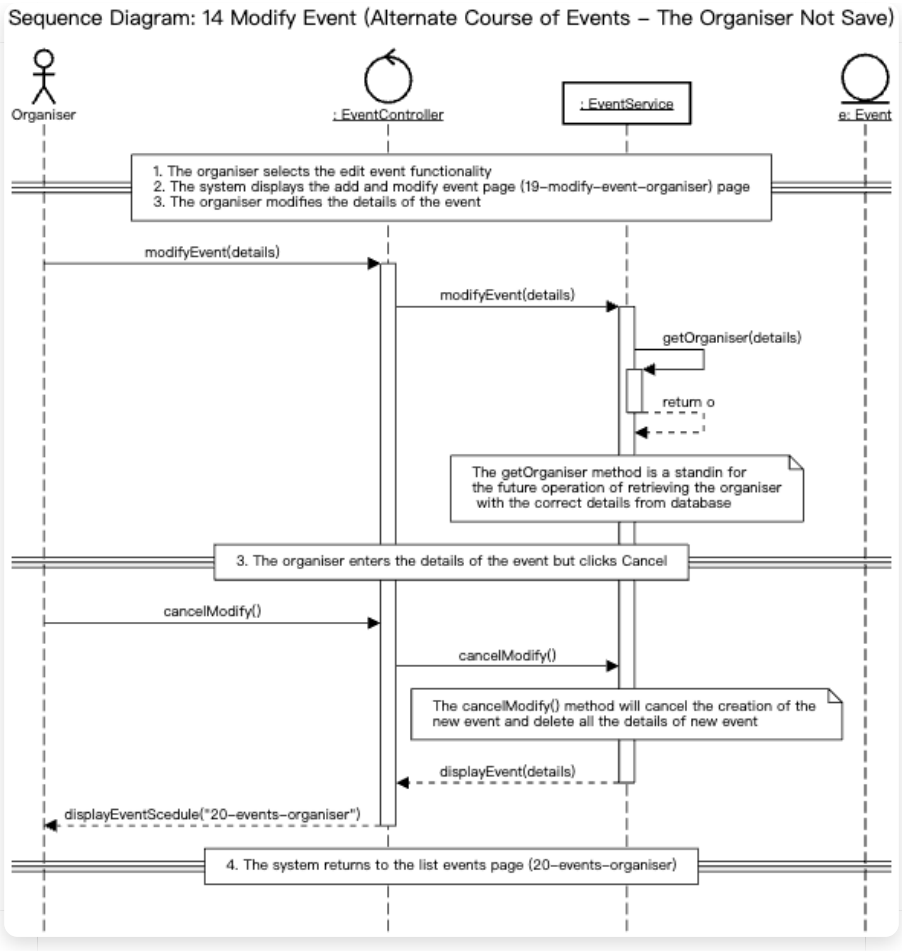
Number:

Use Case 14 - Modify Event

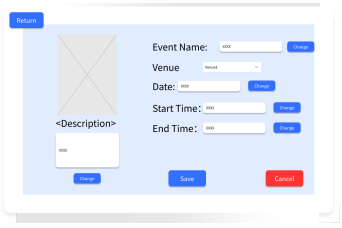
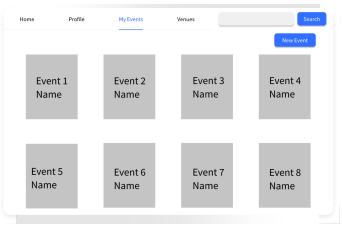
01 - Basic Course of Events



02 - Alternate Course of Events - The Organiser Not Save

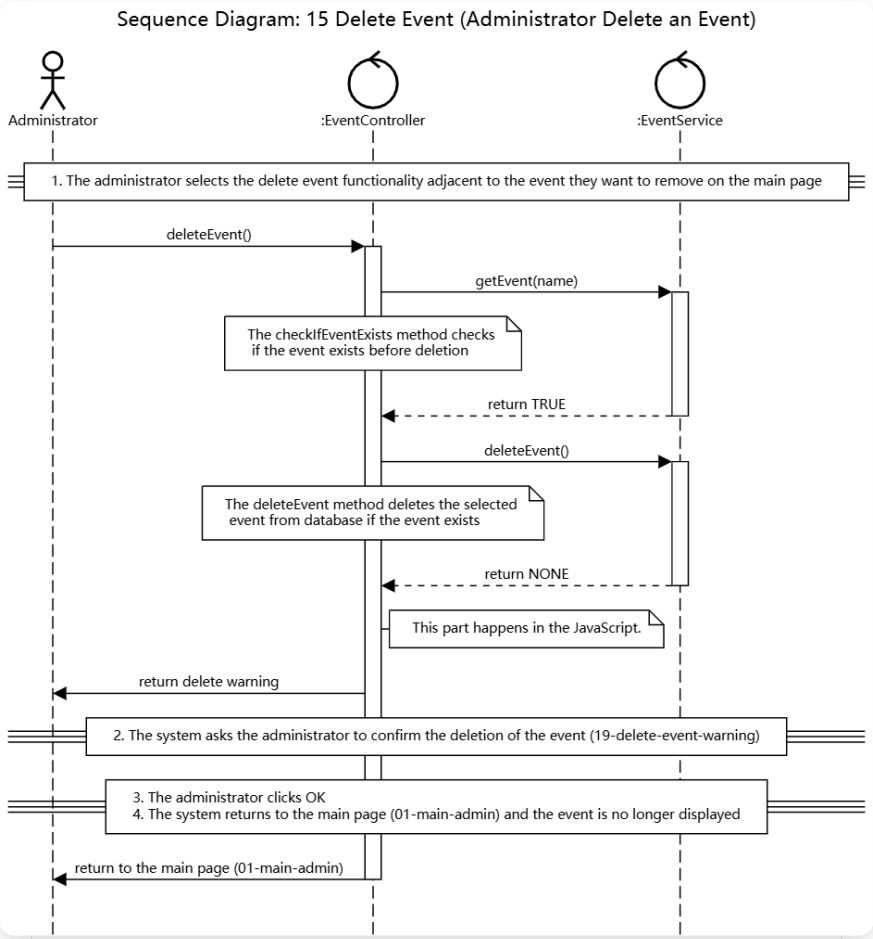


Related UI Prototypes

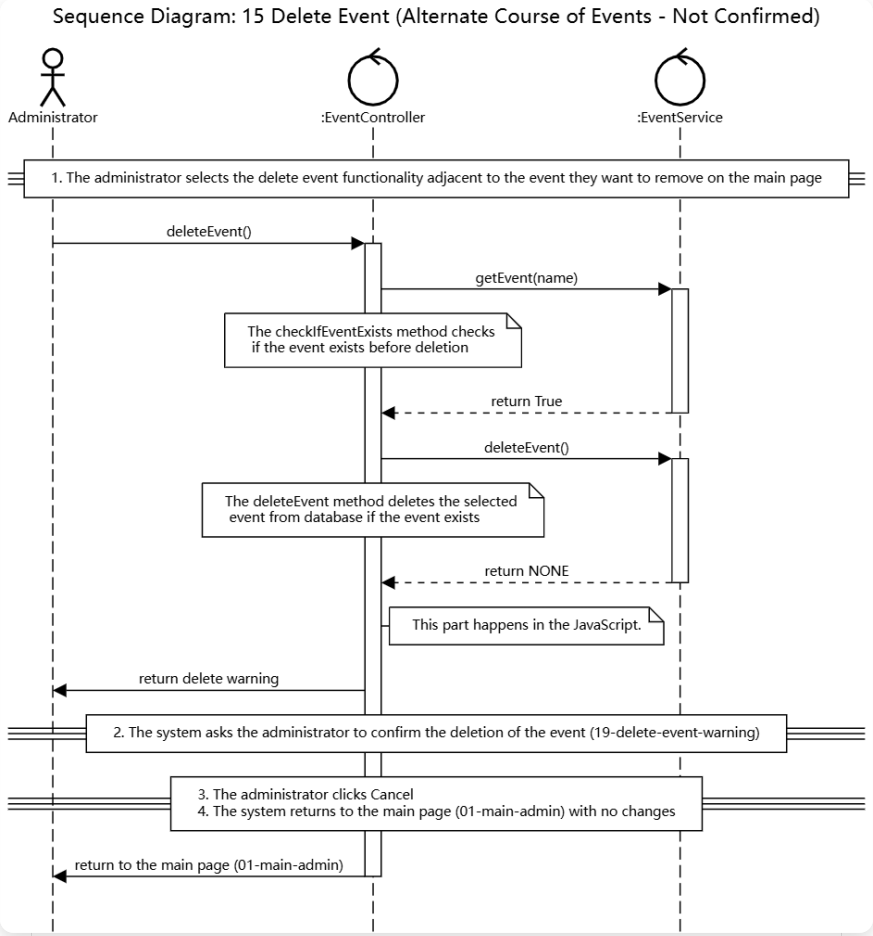
Page Name	Image
The Add and Modify Event Page	
The List Events Page(Organiser)	

Use Case 15 - Delete Event

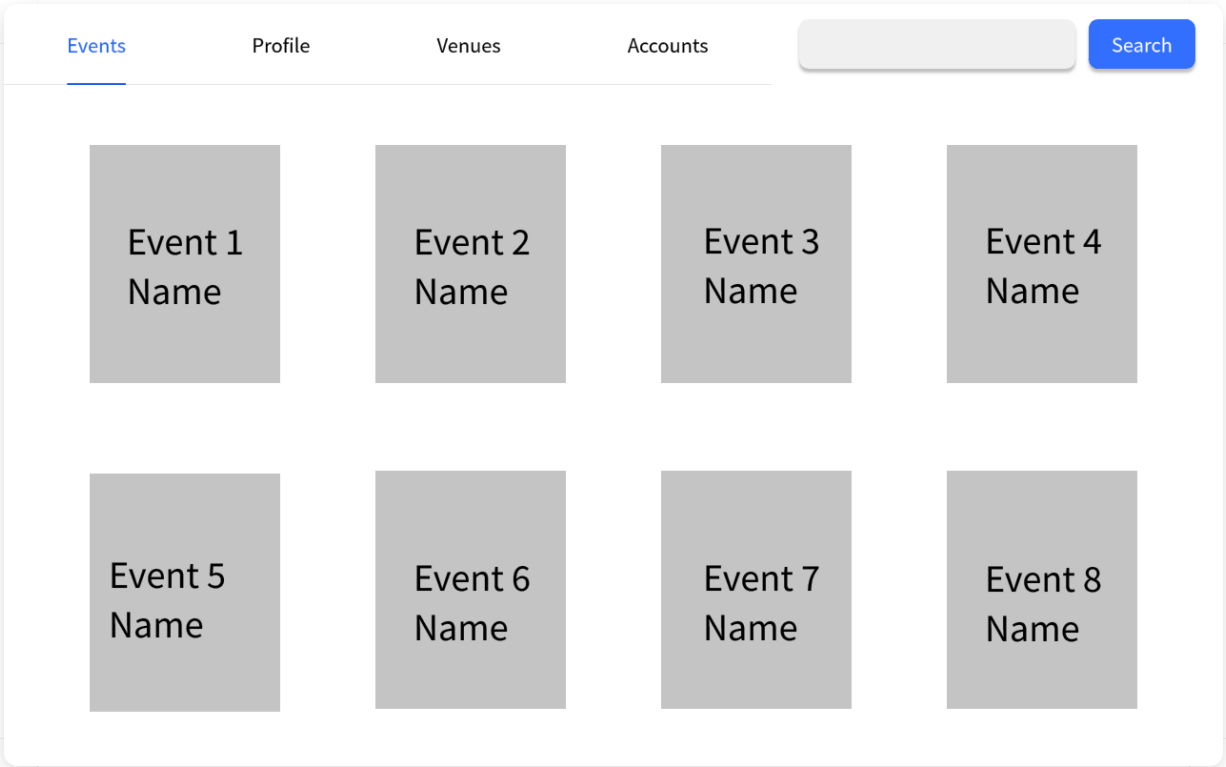
01 - Basic Course of Events



02 - Alternate Course of Events - Not Confirmed



Relevant UI Sketches | Page Name | Image | |———————|———————————————————————————-| | Admin Main Page |

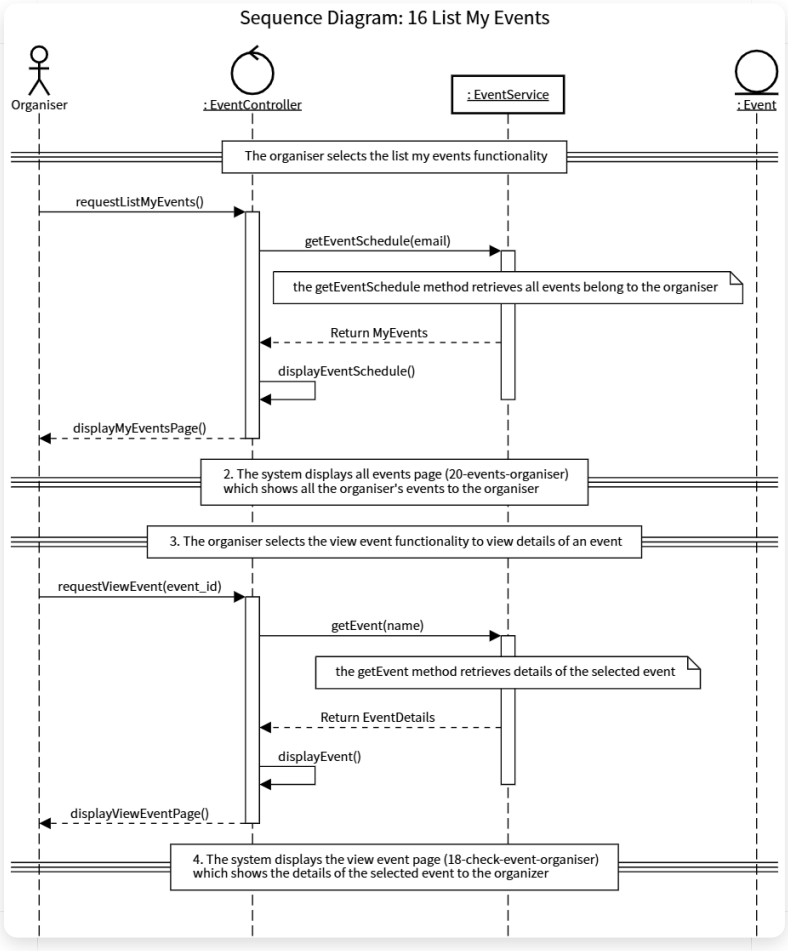


| | Delete Event Warning |

Delete Event Warning

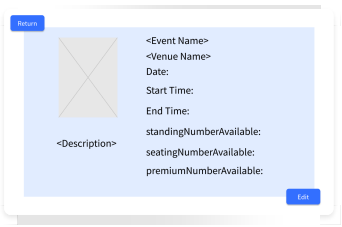
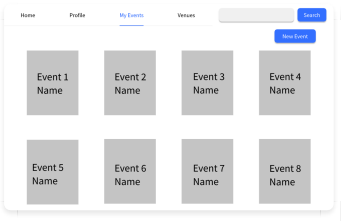
Use Case 16 - List My Events

01 - Basic Course of Events



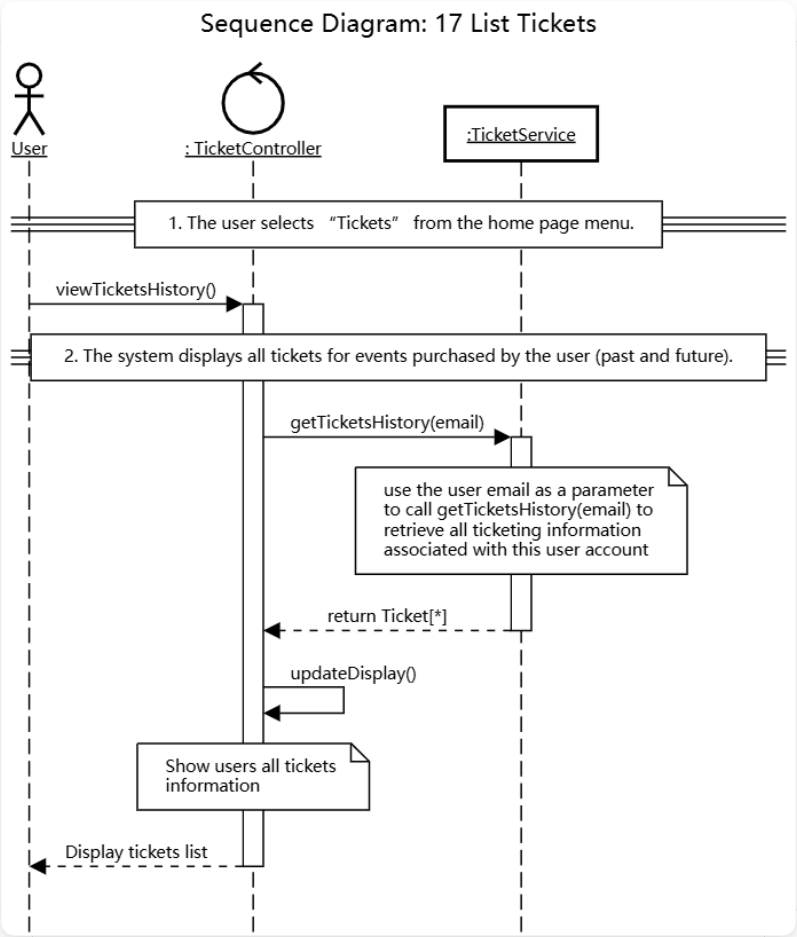
List My Events - Basic Course of Events

Related UI Prototypes

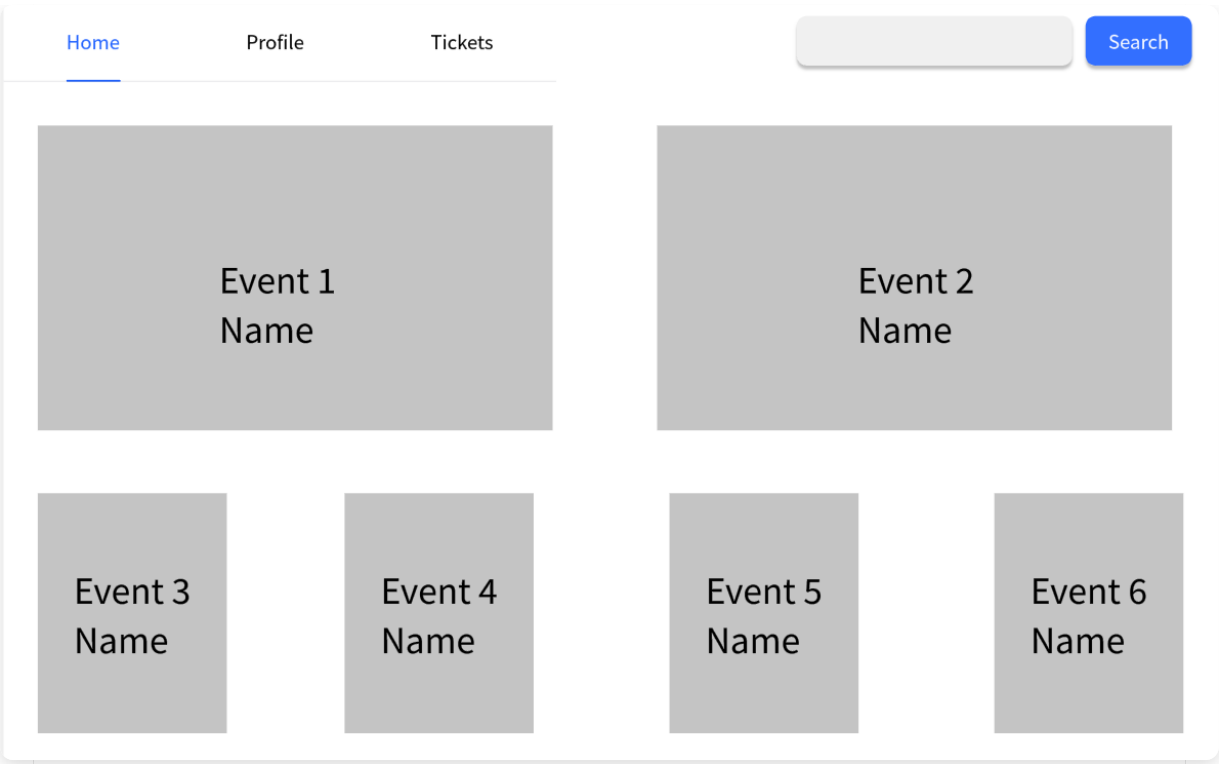
Page Name	Image
18-check-event-organiser	
20-events-organiser	

Use Case 17 - List Tickets

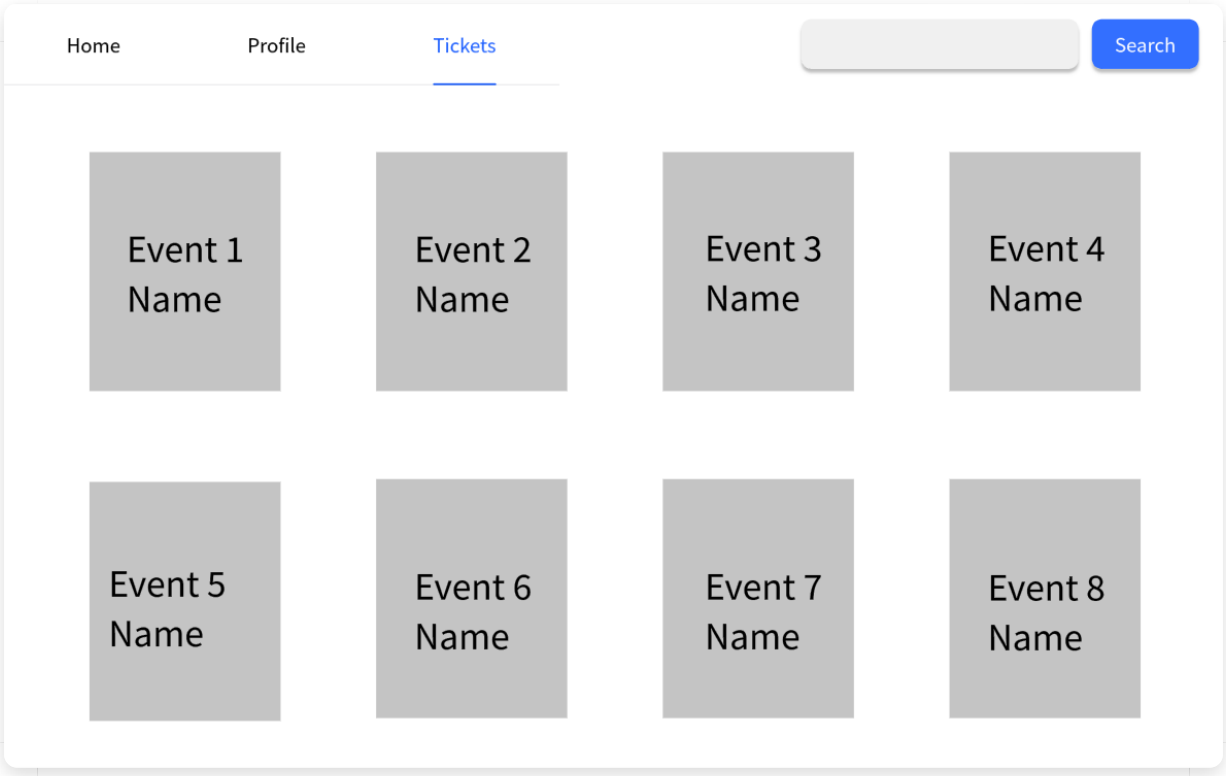
01 - Basic Course of Events



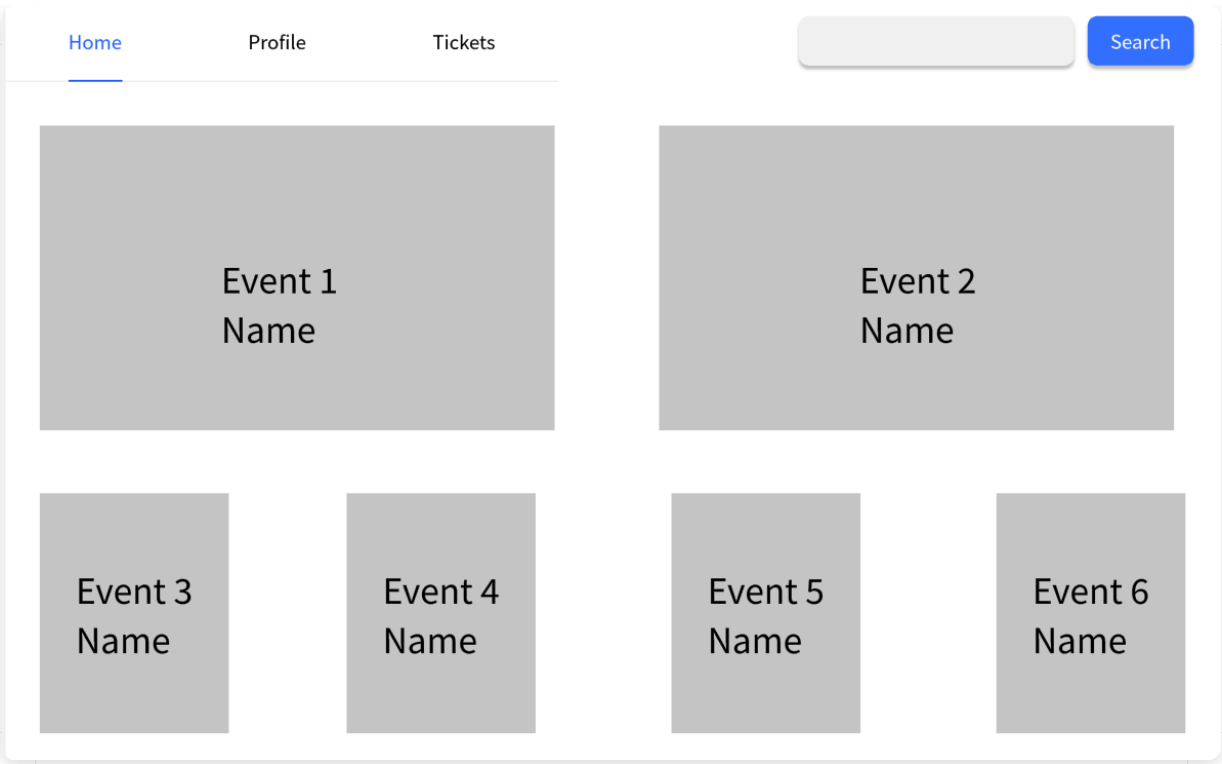
Related UI Prototypes | Page Name | Image | |----------|--------| | Main User Page |



| | Tickets User Page |



|



| | Tickets User Page |