



DESIGN AND INSTALLATION REPORT

RUNNING EVENT SYSTEM

PRESENT TO

Assoc. Prof. Dr. Twittie Senivongse

BY

Ms.Kamolnadda	Dansuputra	5931001021
Ms.Natthawan	Siripokasupkul	5930188521
Ms.Chanissa	Trithipkaiwanpon	5931016421
Mr.Nutchanon	Ploypray	5930166721
Mr.Pisit	Wajanasara	5931042721
Mr.Pawin	Piemthai	5931037621

Systems Analysis and Design

2110332, semester 1/2018

**Department of Computer Engineering,
Faculty of Engineering, Chulalongkorn University**

Contents

List of Illustrations	3-4
List of Tables	4
Part I Project design phase document:	
1. Project name	5
2. Introduction	5
3. Objective of the design document	5
4. Reference	5
5. Terms and definitions	6
6. System design constraint	6-7
7. Design criteria and principles	7-13
8. Class and method design	13-26
9. Human-computer interaction design	27-44
10. Physical architecture design	45-48
11. Impact of the design solution	48-49
Part II Installation and operation phase document:	
1. Conversion plan	49
2. Change management	49-50
3. Post-implementation activities	51

List of illustrations

Figure 1. (a) Class Diagram before using cohesion (b) Class Diagram after using cohesion	3
Figure 2. Class “TransactionList” and class “Transaction” in the transaction design pattern format.	4
Figure 3. Factoring of class “Runner” and class “EventManager” into class “User”	5
Figure 4. Package diagram for Running Event system	7
Figure 5. Revised class diagram in packages	9
Figure 6. Normalized class diagram	14
Figure 7. Layers of software architecture	15
Figure 8. CRC card of Reservation List	16
Figure 9. CRC card of Event	17
Figure 10. CRC card of Transaction	18
Figure 11. Contract specifications of createReservation method	18
Figure 12. Contract specifications of createNewEvent method	19
Figure 13. Contract specifications of createAccount method	19
Figure 14. Method specifications of createReservation method	21
Figure 15. Method specifications of createNewEvent method	22
Figure 16. Method specifications of createAccount method	23
Figure 17. Use case diagram in packages	27
Figure 18. Reserve Running Event Use Scenarios	28
Figure 19. Register New account Use Scenarios	28
Figure 20. Window Navigation Diagram (WMD)	29
Figure 21. Reserve Running Event Use Case Description	30
Figure 22. Manage Event Use Case Description	31
Figure 23. Register New Account	32
Figure 24. Login page for all users	33
Figure 25. Register new account	33
Figure 26. Alert message from duplicated username entered	34
Figure 27. Runner registration page	34
Figure 28. Event manager registration page	35
Figure 29. Overview for runner account	35
Figure 30. Running event search page	36
Figure 31. Running event search result	36
Figure 32. Running event information page	37
Figure 33. Reserve running event page	37
Figure 34. Alert message from invalid information entered	38
Figure 35. Valid information for reserve running event page	38
Figure 36. Payment method for reserve running event	39
Figure 37. Alert message from invalid payment information entered	39
Figure 38. Confirmation pop-up for payment	40
Figure 39. Reservation succeeded message	40

<i>Figure 40. Overview for event manager account</i>	41
<i>Figure 41. Manage event page</i>	41
<i>Figure 42. View participants page</i>	42
<i>Figure 43. All changes saved information message</i>	42
<i>Figure 44. Edit running event page</i>	43
<i>Figure 45. All changes saved information message</i>	43
<i>Figure 46. Deployment diagram</i>	44
<i>Figure 47. Network diagram</i>	45

List of Tables

<i>Table 1. Terms and definitions</i>	1
<i>Table 2. Detailed descriptions for package diagram</i>	6
<i>Table 3. UI design principles</i>	28
<i>Table 5. Nonfunctional requirements and physical architecture design</i>	46
<i>Table 6. Hardware and software specification</i>	47

Part I Project design phase document

1. Project name: Running Event System

2. Introduction

For the past several years, running events has become trendy in Thailand. Men and Women, young and old, famous and ordinary people are all interested in these events. Almost every month, there are two or three running events. This is a good phenomenon for Thai people since it encourages people to exercise more and care more for their health.

However, since these running events have just become popular in Thailand, they still don't have an officially centralized system for event organizers and runners to communicate. Consequently, there are many problems and inconvenience such as events are announced separately so it is hard for runners to look for the events that is most proper to them, event managers find it hard to promote and manage their event, runners don't receive enough information of the running event and difficult for runners to find and pay for the reservation etc.

Our team concern with all the problems stating above so we decide to create a centralized platform for runners and event organizers to communicate. We'll also include useful functions to enhance user's convenience and hope that this project will encourage more people to go to running events.

3. Objective of the design document

The objective of the design document is to describe how the Running Event System will be constructed based on the followed activities.

1. Defining system design constraint
2. Design criteria and principles
3. Class and method design
4. Designing the human-computer interface
5. Physical architecture design
6. Providing the impact of the design solution

4. Reference

1. Running Event System Project Proposal : Revised Edition
2. Running Event System Project Analysis : Revised Edition

5. Terms and definitions

Term	Definition
Running event	A casual running event for amateur runners and normal people.
Event manager	A group of people who organizes the running event.
Running type	A type of running; This system will focus on only two: road running and cross country running.
Road running	The sport of running on a measured course over an established road.
Cross country running	a sport in which teams and individuals run a race on open-air courses over natural terrain.
Running distance	A total running distance over the entire course: usually ranging from short run (2.5 km) to Marathon (42 km).
User	Typical user of the system; This includes both interested runners and event managers.
Community	Refer to runners community
Reservation cost	A cost that runners must pay in advance to the event managers in order to participate in the running event.
Runner	A group of people who is interested in running event.

Table 1. Terms and definitions

6. System design constraint

The objective of this topic is explain the limitation and restriction of the system which should be concerned in the design process.

1. System's platform
 - 1.1. The system is a web platform system. It requires users to have access to the internet with mobile or desktop devices using mainstream web browser such as Microsoft Edge, Mozilla Firefox, Apple Safari, Opera Browser and Google Chrome.
 - 1.2. The system required HTTPS and WebSocket as a communication protocol between server and clients.
2. User capabilities and limitations
 - 2.1. Before using the system both runner and event manager must register to the system as their appropriate role.
 - 2.2. Runner must have credit/debit card for reservation payment.
 - 2.3. The reservation payment have to be done using the system payment gateway only, any payment that had done using other ways are not responsible by the system.
3. Security
 - 3.1. The system must only allow appropriate user to access a system function.

- 3.2. User's private information such as credit/debit card information must only be visible to the user.
- 3.3. User's password must be stored as a hashed value.
- 4. Available Time
 - 4.1. The system analysis and design specification must be finished with in December 2018.
- 5. Physical size limitation
 - 5.1. The storage spaces available for each user are 500 MiB.
- 6. Cultural constraint
 - 6.1. The running event in the web platform must be located Thailand.

7. Design criteria and principles

The objective of this topic is to transform functional and non-functional requirements which is in a form of natural language, feasibility analysis and constraints into overall design of the system.

To design new Class Diagram from SRS document, we use design criteria and principles as follows.

- 1. Layers
- 2. Design criteria: Coupling and Cohesion
- 3. Adding specifications
- 4. Identifying opportunities to reuse classes
- 5. Restructuring the Design: Factoring and Normalization
- 6. Partitions: Package Diagram
- 7. Optimizing the design

7.1 Layers

We use layers to represent and separate elements of the software architecture to ease understanding of a complex system. In our system 5 layers which are Foundation layer, Problem Domain layer, Human-Computer Interaction later, Data Management layer and Physical Architecture layer have been used.

- 1. Problem Domain layer: This layer corresponds to all main logics that the main systems needed. It consists of class packages which are responsible for receiving inputs and representing outputs of the system. In our system, there are 7 user case packages from the use case diagram so that there are 7 major packages in the layer.
- 2. Human-Computer Interaction layer: Classes are created in a form of Form which interacts to the human users, for receiving user input and send to the Problem Domain Layer and vice versa.
- 3. Physical Architecture layer: This layer deal with communication between operating system, software and network of the system.

4. Data Management layer: This layer corresponds to application data related logic and manipulation. Its responsibility is to provide database services for Problem Domain layer and other layer that required database access.
5. Foundation layer: This layer includes classes, data types and utilities that are necessary for any system such as integer, string and collections.

7.2 Design criteria

We use design criteria to make our design maintainable for changing in later version , so we use low coupling and high cohesion on this design to reduce dependencies between modules and classes by using Law of Demeter.

1. Coupling: We decrease interdependency between classes to prevent changes in the design when one object has changed. However, the revision of class diagram does not need to change because there is no coupling in each class.
2. Cohesion: As we reviewed the class diagram, we found that CalendarController class and EventController class are do the same roles as controlling event. As a result, we decide to put methods from CalendarController into EventController as show in *Figure 1*.

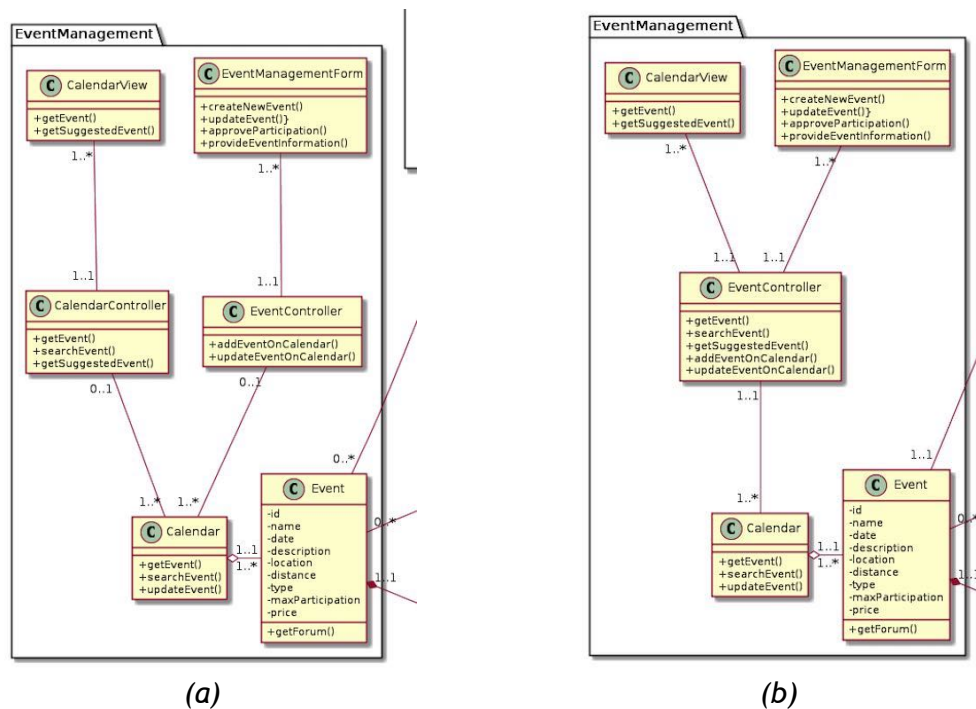


Figure 1. (a) Class Diagram before using cohesion (b) Class Diagram after using cohesion

7.3 Adding Specifications

After applying design criteria, we review our current analysis models. We found that our analysis models have sufficient and necessary classes to solve the problem of our system, no missing attributes or methods, no extra or unused attributes or methods and no missing or extra classes.

Next, we decide on the method signatures and constraints which will be consider later on contracts specification.

7.4 Identifying opportunities to reuse classes

In this topic, we try to identify the opportunities for reusing the classes. In the development team consideration, the class diagram has proper functionalities and easy to understand. Moreover, we have already use design patterns for our class design as described below.

Transaction Design Pattern: We use transaction design pattern for a class that required to hold a collection of object, for example class “TransactionList” and class “Transaction” satisfied this design pattern as shown in *Figure 2*.

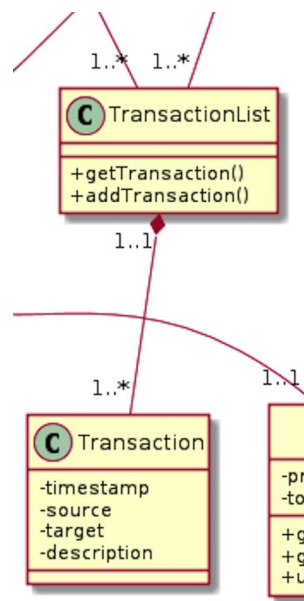


Figure 2. Class “TransactionList” and class “Transaction” in the transaction design pattern format.

7.5 Restructuring the Design: Factoring and Normalization

In this topic, we restructure the design to simplify the design and also identify missing classes by using these following topics.

1. Factoring: As we reviewed the class diagram, we had already used this technique in our class diagram. For example, we created a detailed User class from Runner class and Event Manager class to make the design easier to understand and sustainable, as shown in *Figure 3*.

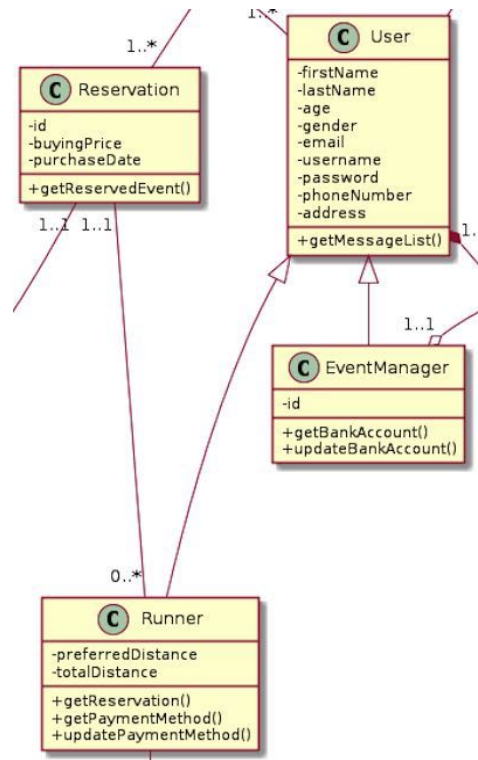


Figure 3. Factoring of class “Runner” and class “EventManager” into class “User”

2. Normalization: We use 2 types of normalization to the design.
 - 2.1. Convert association classes to normal classes
 - 2.2. Convert association and aggregation to attributes

7.6 Partitions: Package Diagram

After the design is easy to understand, we group classes into packages based on use case diagram. So, we divide them into 7 packages.

1. Event management package
2. Reserve event package
3. Account package
4. Communication package
5. Log system package
6. Payment system package

Detailed descriptions are shown in the following table.

Package	Contained Classes	Description
Event management package	<ul style="list-style-type: none"> - CalendarView - EventManagement Form - EventController - Calendar - Event 	The package contains classes associated with event management for event managers to interact with.
Reserve event package	<ul style="list-style-type: none"> - ReservationForm - ReservationController - ReservationList - Reservation 	The package contains classes associated with reserve event for runners to interact with.
Account package	<ul style="list-style-type: none"> - CitizenVerificationHandler - EmailDistributionHandler - AccountForm - AccountController - UserList - User - Runner - EventManager - BankAccount - PaymentMethod 	The package contains classes associated with account including account management such as AccountController, AccountForm and CitizenVerificationHandler etc.
Communication package	<ul style="list-style-type: none"> - ForumView - ForumController - Forum - Post - MessageView - MessageController - MessageList - Message 	The package contains classes associated with communication and messages for interaction with event management and account packages
Log system package	<ul style="list-style-type: none"> - SysAdminView - LogController - SysAdminController - LogList - TransactionList - Log - Transaction 	The package contains classes associated with log system for system admin to monitor about what happens in the system.
Payment system package	<ul style="list-style-type: none"> - PaymentHandler - PaymentController 	The package contains classes associated with payment system.

Table 2. Detailed descriptions for package diagram

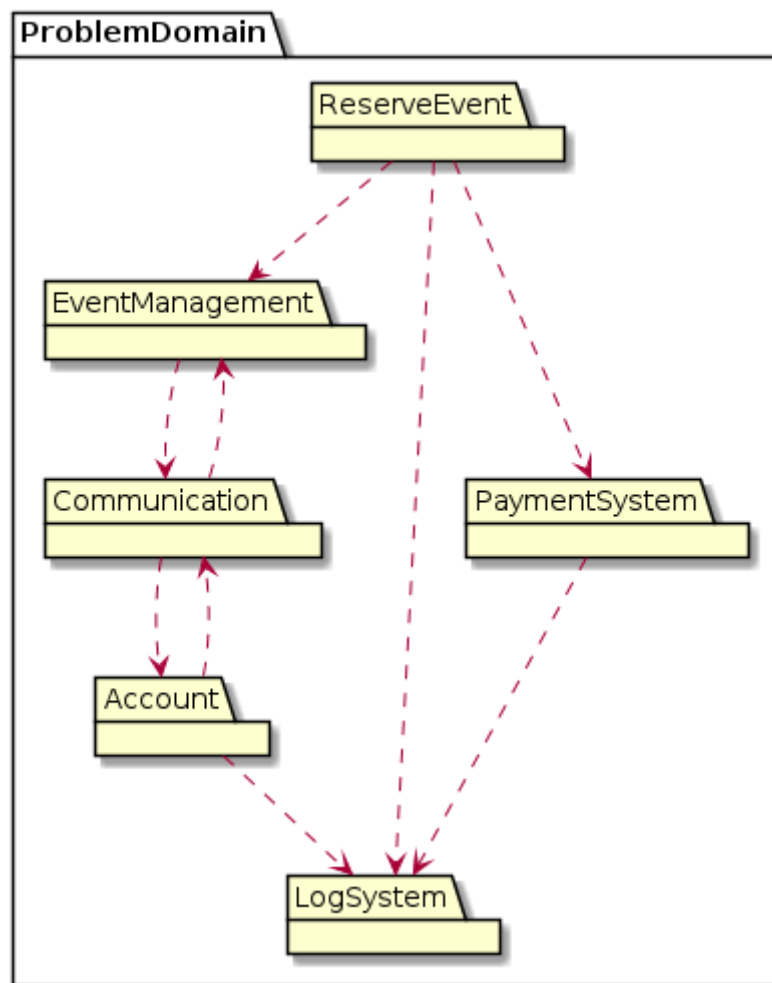


Figure 4. Package diagram for Running Event system

7.7 Optimizing the design

After making the design is easy understandability and group classes into packages, we use these criteria to optimize the design to improve efficient of the system by as follows.

1. Review access path between objects: As our class diagram is based on Entity-Control-Boundary pattern (ECB) and long path call is based on path of ECB. Since we have to keep structure of ECB, we decided not to do any change on these path.
2. Review each attribute of each class: We reviewed each attribute of each class and we found that all attributes are in appropriate classes.
3. Review direct and indirect fan-out of each method: After make an analysis, amount of fan-out of each class is in an acceptable amount and no change has been made.
4. Consider execution order of statements in often-used methods to arrange them for efficiency: Our class diagram is already in the Entity-Control-Boundary pattern so the execution order is satisfied by this pattern and we decided not to make any change.

5. Avoid re-computation by creating derived attributes and triggers: After we analyze the class diagram, we found that there are no re-computation problems in our class diagram and the current diagram is acceptable.
6. Consider combining classes that form one-to-one association: After we analyze the class diagram, we found that there is no one-to-one association in the class diagram, so it passed this criteria.

8. Class and method design

Class and method design is an important thing to make the design easy to understand and help programmer implement the system easily. Therefore, from the previous topic that using design criteria and principles, the class diagram of Running event system has a result as follows.

8.1 Revised class diagram in packages

After revising the class diagram, the design classes have already had low coupling and high cohesion as we expected after applying design criteria and principles in Topic 7. Moreover, the class diagram is maintainable for changes in later version, for example, if PaymentMethod class is going to change in the later version, it will only effect on Runner class and itself.

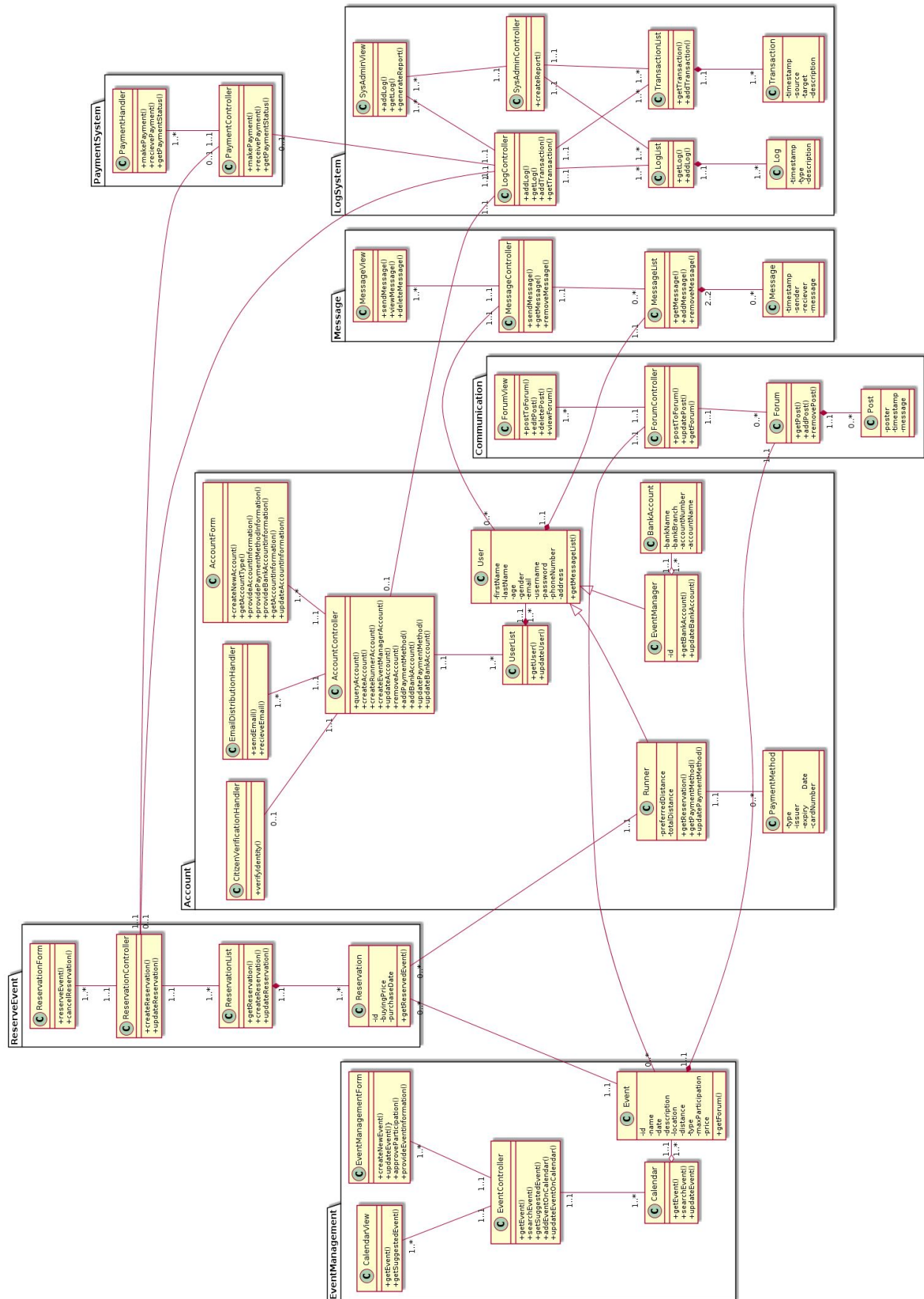


Figure 5. Revised class diagram in packages

8.2 Normalized class diagram

Normalized class diagram is the design class that make programmer easier to implement in implementation phase and can identify missing classes, attributes and methods easily. According to restructuring the design by normalization, the design has results as shown in *Figure 6*.



Figure 6. Normalized class diagram

8.3 Package diagram

We separate elements of the software architecture into 5 layers. Also, we used partitions, according to 7.6, to give more understanding to overall architecture of the system.

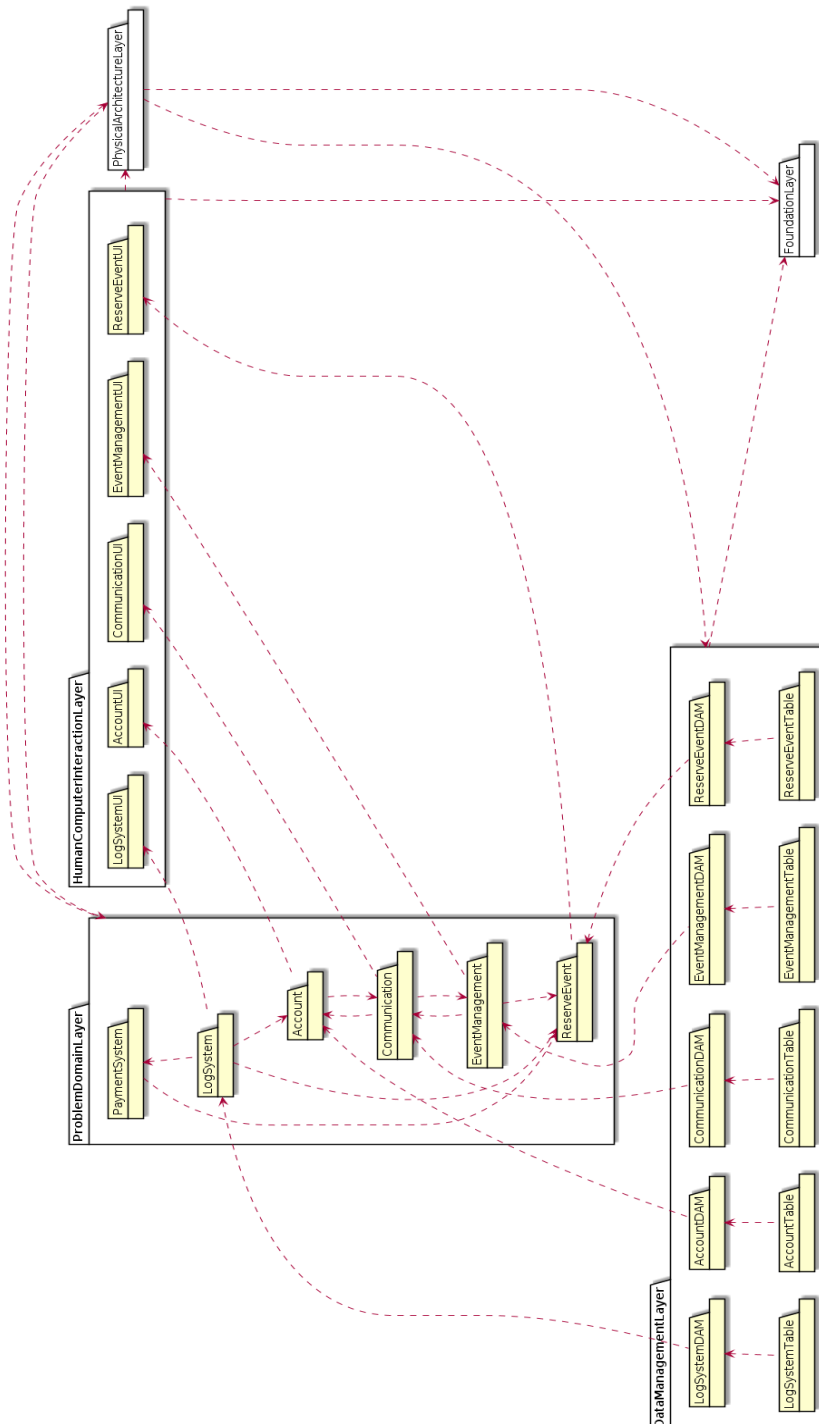


Figure 7. Layers of software architecture

8.4 CRC cards with invariants

According to 7.3, we added constraints and invariants to classes during design and revision process. Examples of the result is shown below.

Front:

Class Name: Reservation List	ID: 1	Type: Concrete, Domain
Description: A list of reservations which create by runners		Associated Use Cases: 1
Responsibilities createReservation(id, buyingPrice, purchaseDate, list, event, reserver) updateReservation(id, buyingPrice, purchaseDate, list, event, reserver) getReservation()	Collaborators Reservation Reservation Reservation	

Back:

Attributes: -reservation[1..*] : Reservation -controller[1..1] : ReservationController
Relationships: Generalization (a-kind-of): - Aggregation (has-parts): Reservation[1..*] Other Associations: Reservation Controller[1..1]

Figure 8. CRC card of Reservation List

Front:

Class Name: Event	ID: 2	Type: Concrete, Domain
Description: A running event created by event manager		Associated Use Cases: 2
Responsibilities getForum()	Collaborators Forum	

Back:

Attributes: -id[1..1] : unsigned long -name[1..1] : String -date[1..1] : Date -description[1..1] : String -location[1..1] : String -distance[1..1] : double {distance >= 0} -type[1..1] : Enum -maxParticipation[1..1] : Integer {maxParticipation >= 0} -price[1..1] : double -forum[1..1] : Forum -calendar[1..1] : Calendar -forumController[1..1] : ForumController -reservation[0..*] : Reservation
Relationships: Generalization (a-kind-of): - Aggregation (has-parts): Reservation[0..*], Forum[1..1], Calendar[1..1] Other Associations: Forum Controller[1..1]

Figure 9. CRC card of Event

Front:

Class Name: Transaction	ID: 3	Type: Abstract, Domain
Description: A form of confirmation from payment		Associated Use Cases: 1, 1-5
Responsibilities -	Collaborators -	

Back:

Attributes: -timestamp[1..1] : Timestamp -source[1..1] : String -target[1..1] : String -description[0..1] : String -list[1..1] : TransactionList
--

Relationships: Generalization (a-kind-of): - Aggregation (has-parts): Transaction list[1..1] Other Associations: -

Figure 10. CRC card of Transaction

8.5 Contract specifications with pre and post condition

Contract specifications specify precondition and postcondition which useful for system analysis and development process.

Method Name: createReservation	Class Name: ReservationController	ID: 25
Clients: ReservationForm		
Associated Use Cases: Reserve Running Event		
Description of Responsibilities: Implement the necessary behavior to reserve a new running event. This method recieves an event object and performs necessary step to add the reserver to the participant list in a waiting for approval state.		
Arguments Received: buyingPrice:double, anEvent:Event, aReserver:Runner		
Type of Value Returned: void		
Preconditions: aReservation.getReservedEvent() = anEvent and not ReservationList.includes(aReservation)		
Postconditions: aReservation.getReservedEvent() = anEvent and ReservationList = ReservationList@pre.including(aReservation)		

Figure 11. Contract specifications of createReservation method

Method Name: createNewEvent	Class Name: EventManagerForm	ID: 34
Clients: EventController		
Associatated Use Cases: Manage Event		
Description of Responsibilities: This method receives a call from user and operates each step to create the new event.		
Arguments Received: -		
Type of Value Returned: anEvent:Event		
Preconditions: not Calendar.including(anEvent)		
Postconditions: Calendar = Calendar@pre.including(anEvent)		

Figure 12. Contract specifications of createNewEvent method

Method Name: createAccount	Class Name:AccountController	ID: 35
Clients:AccountForm		
Associatated Use Cases: Register New Account		
Description of Responsibilities: This method receives a call from user and operates each step to create a new account.		
Arguments Received:username, password		
Type of Value Returned: void		
Preconditions: not UserList.including(User)		
Postconditions:UserList = UserList@pre.including(User)		

Figure 13. Contract specifications of createAccount method

8.6 Method specifications

The purpose of method specifications is to define the description of each method which can make the development process more intuitive and more robust.

Method Name: createReservation	Class Name: ReservationList	ID: 25
Contract ID: 25	Programmer: Pawin Piemthai	Date Due: 25 November, 2018
Programming Language: Javascript		
Triggers/Events: Runner wants to reserve a running event.		
Argument Received: Data Type:	Notes:	
buyingPrice:double		
anEvent:Event		
aReserver:Runner		
Message Sent & Arguments Passed: Class Name.MethodName:	Data Type:	Notes:
reserver.getReservation()		
Reservation.new()		
Runner.getPaymentMethod()		
PaymentController.makePayment()		
ReservationList.createReservation()		

EmailDistributionHandler.sendEmail()		
LogController.addLog()		
Argument Returned: Data Type:	Notes:	
void		
Algorithm specification: reservationList = reserver.getReservation() reservation = new Reservation() reservation.buyingPrice = buyingPrice reservation.purchaseDate = Date.currentDate() reservation.event = anEvent reservation.reserver = aReserver Reservation.status = WAITING_APPROVAL if (anEvent.price > 0) then if (aReserver.getPaymentMethod == None) then AccountController.addPaymentMethod(reserver) payment = reserver.getPaymentMethod() status = PaymentController.makePayment(payment, buyingPrice) if (status == PAYMENT_OK) then reservationList.createReservation(reservation) EmailDistributionHandler.sendEmail(reserver.email, confirmMessage) LogController.addLog(RESERVATION_COMPLETE, description)		
Misc. Notes: none		

Figure 14. Method specifications of createReservation method

Method Name: createNewEvent	Class Name: EventManagerForm	ID: 50
Contract ID: 34	Programmer: Chanissa Trithipkaiwanpon	Date Due: 25 Nov 2018
Programming Language: Javascript		
Trigger/Events: Event managers creates a new event on our website.		
Arguments Received: -		
Data Type:	Notes:	
Message Sent & Arguments Passed:		
ClassName.MethodName	Data Type:	Notes:
EventController.addEventOnCalendar()		
Argument Returned:		
Data Type:	Notes:	
anEvent:Event		
Algorithm Specification: EventController.addEventOnCalendar()		
Misc. Notes:-		

Figure 15. Method specifications of createNewEvent method

Method Name: createAccount	Class Name: AccountController	ID: 4
Contract ID: 35	Programmer: Nutchanon Ploypray	Date Due: 25 NOV 2018
Programming Language: Javascript		
Trigger/Events: Users create an account on our website.		
Arguments Received:		
Data Type:	Notes:	
username: String	A spetic name for each user for using our website.	
password: String	A number that is used for authentication for its username.	
Message Sent & Arguments Passed:		
ClassName.MethodName	Data Type:	Notes:
AccountForm.getAccountType()	String	
AccountForm.createRunnerAccount()		
AccountForm.createEventManagerAccount()		
Argument Returned:		
Data Type:	Notes:	
void		
Algorithm Specification: If (AccountForm.getAccountType() == “Runner”) AccountController.createRunnerAccount() else if (AccountForm.getAccountType() == “Event manager”) AccountController.createEventManagerAccount()		
Misc. Notes:-		

Figure 16. Method specifications of createAccount method

8.7 Verifying and validating class and method design

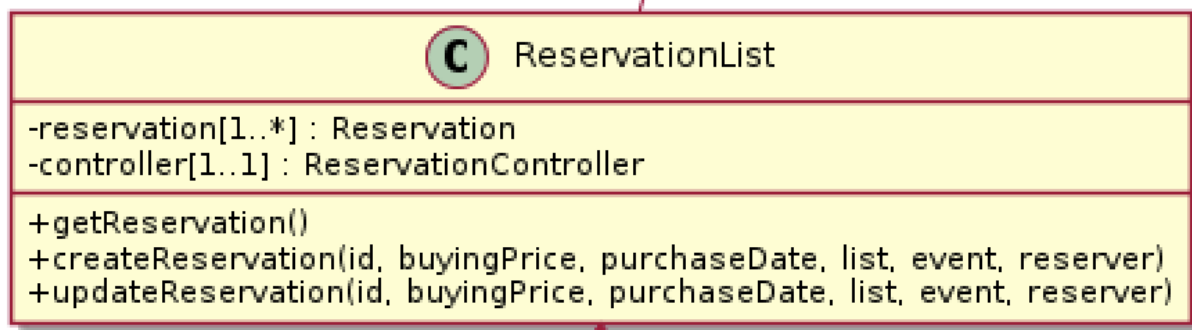
1. All attributes in class diagram are mentioned in CRC cards. For example, all attributes in ReservationList class diagram are mentioned in the CRC card of it.

Front:

Class Name: Reservation List	ID: 1	Type: Concrete, Domain
Description: A list of reservations which create by runners		Associated Use Cases: 1
Responsibilities <u>createReservation(id, buyingPrice,</u> <u>purchaseDate, list, event, reserver)</u> <u>updateReservation(id, buyingPrice,</u> <u>purchaseDate, list, event, reserver)</u> <u>getReservation()</u>		Collaborators Reservation Reservation Reservation

Back:

Attributes: -reservation[1..*] : Reservation -controller[1..1] : ReservationController
Relationships: Generalization (a-kind-of): - Aggregation (has-parts): Reservation[1..*] Other Associations: Reservation Controller[1..1]



2. All arguments received in contract specification are mentioned in arguments received in method specification. For example, all arguments received in contract specification of createReservation are mentioned in arguments received in createReservation method specification.

Method Name: <u>createReservation</u>	Class Name: <u>ReservationController</u>	ID: 25
Clients: <u>ReservationForm</u>		
Associated Use Cases: Reserve Running Event		
Description of Responsibilities: Implement the necessary behavior to reserve a new running event. This method <u>receives</u> an event object and performs necessary step to add the <u>reserver</u> to the participant list in a waiting for approval state.		
Arguments Received: <u>buyingPrice:double</u> , <u>anEvent:Event</u> , <u>aReserver:Runner</u>		
Type of Value Returned: void		
Preconditions: <u>aReservation.getReservedEvent()</u> = <u>anEvent</u> and not <u>ReservationList.includes(aReservation)</u>		
Postconditions: <u>aReservation.getReservedEvent()</u> = <u>anEvent</u> and <u>ReservationList = ReservationList@pre.including(aReservation)</u>		

Method Name: <u>createReservation</u>	Class Name: <u>ReservationList</u>	ID: 25
Contract ID: 25	Programmer: <u>Pawin Piemthai</u>	Date Due: 25 November, 2018
Programming Language: Javascript, HTML, CSS		
Triggers/Events: Runner wants to reserve a running event.		
Argument Received: Data Type:	Notes:	
<u>buyingPrice:double</u>		
<u>anEvent:Event</u>		
<u>aReserver:Runner</u>		
Message Sent & Arguments Passed: Class Name.MethodName:	Data Type:	Notes:

3. All arguments received in contract specification are enough to create class diagram. For example, all arguments received in contract specification of createNewEvent are enough for its class diagram. For attribute of class diagram that does not have in arguments received, it can create by getting from other class or using other tools.

Method Name: <u>createNewEvent</u>	Class Name: <u>EventManagerForm</u>	ID: 34
Clients: <u>EventController</u>		
Associated Use Cases: Manage Event		
Description of Responsibilities: This method receives a call from user and operates each step to create the new event.		
Arguments Received: -		
Type of Value Returned: <u>anEvent:Event</u>		
Preconditions: not Calendar.including(<u>anEvent</u>)		
Postconditions: Calendar = Calendar@pre.including(<u>anEvent</u>)		

Method Name: <u>createNewEvent</u>	Class Name: <u>EventManagerForm</u>	ID: 50
Contract ID: 34	Programmer: Chanissa Trithipkaiwanpon	Date Due: 25 Nov 2018
Programming Language: Javascript		
Trigger/Events: Event managers creates a new event on our website.		
Arguments Received: -		
Data Type:	Notes:	
Message Sent & Arguments Passed:		
ClassName.MethodName	Data Type:	Notes:
<u>EventController.addEventOnCalendar()</u>		
Argument Returned:		
Data Type:	Notes:	
<u>anEvent:Event</u>		
Algorithm Specification: <u>EventController.addEventOnCalendar()</u>		
Misc. Notes:-		

9. Human-computer interaction design

This section describes the design of the system. With regard to the actual use and user behavior to create the most satisfying system. And documents prepared for use in communication and synchronization between developers to create a system. The developers can create a system that matches the design.

1. Use case diagram in packages

Overview of the running event management system which contains all functional requirement in the pattern of use case diagram showing the dependency for each use case with other use case and actors.

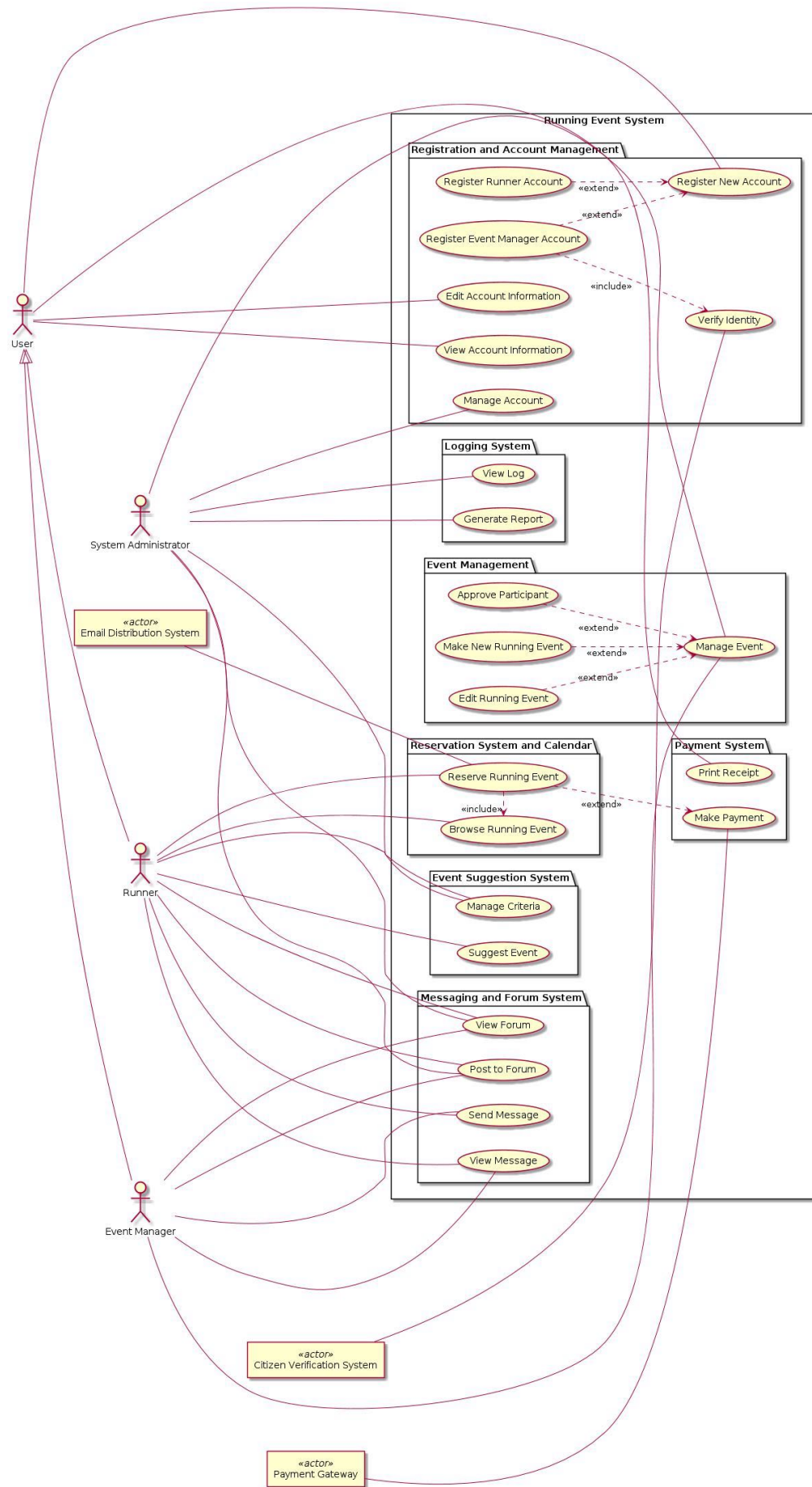


Figure 17. Use case diagram in packages

2. Use Scenarios

Use Scenarios are used to explain the steps performed by users to accomplish some part of their work and it is one path through an essential use case.

Use Scenarios : Reserve Running Event
<ol style="list-style-type: none"> 1. User login using username and password 2. User search for a running event 3. User select a desired running event 4. User apply his/her information for the reservation 5. User pay for the reservation

Figure 18. Reserve Running Event Use Scenarios

Use Scenarios : Register New Account
<ol style="list-style-type: none"> 1. User click register in the login page 2. User fill in desired username and password 3. User choose between register as runner or register as event manager 4. User fill in required personal information 5. User click register

Figure 19. Register New account Use Scenarios

3. User interface design principles

System User Interface (UI) are designed following this principle to make the system perform better user experience.

UI design principles

Principle	How to apply with the user interface design
Layout	Split the screen into 2 section, navigation section and body section. The navigation section show the user where they are and body section is the content of the page
Content Awareness	User always knows where they are by the navigation bar and textboxes in reservation form are classify in group. Also, titles of all interfaces and labels of fields within each area is included
Aesthetic	Use white space to separate each section, Use minimalist design to make website look simple and easy to read, Avoid italics and underlining. Text is not to crowded.
User Experience	The system is easy to use. User can use without learning. The interface is easy to understand.
Consistency	All part of our system work in the same way. Form and reports has the same descriptors
Minimal User Effort	Command that is used frequently is on the first page, use 3 click rule to minimize effort to finish task

Table 3. UI design principles

The UI respond to our non-functional requirement in term of learnability,useability and maintainability.

Learnability - Our System interface is easy to learn how to use.

Useability - Every task is finished in 3 click rules and.

Maintainability - Our UI have the same pattern and work in the same way.

4. Window Navigation Diagram (WND)

We create window navigation diagram to show the overview of all screens,and other components used by the system and how the user will moves from one screen to another.

We provide two window navigation diagram for our main features of our system.

Overall window navigation diagram

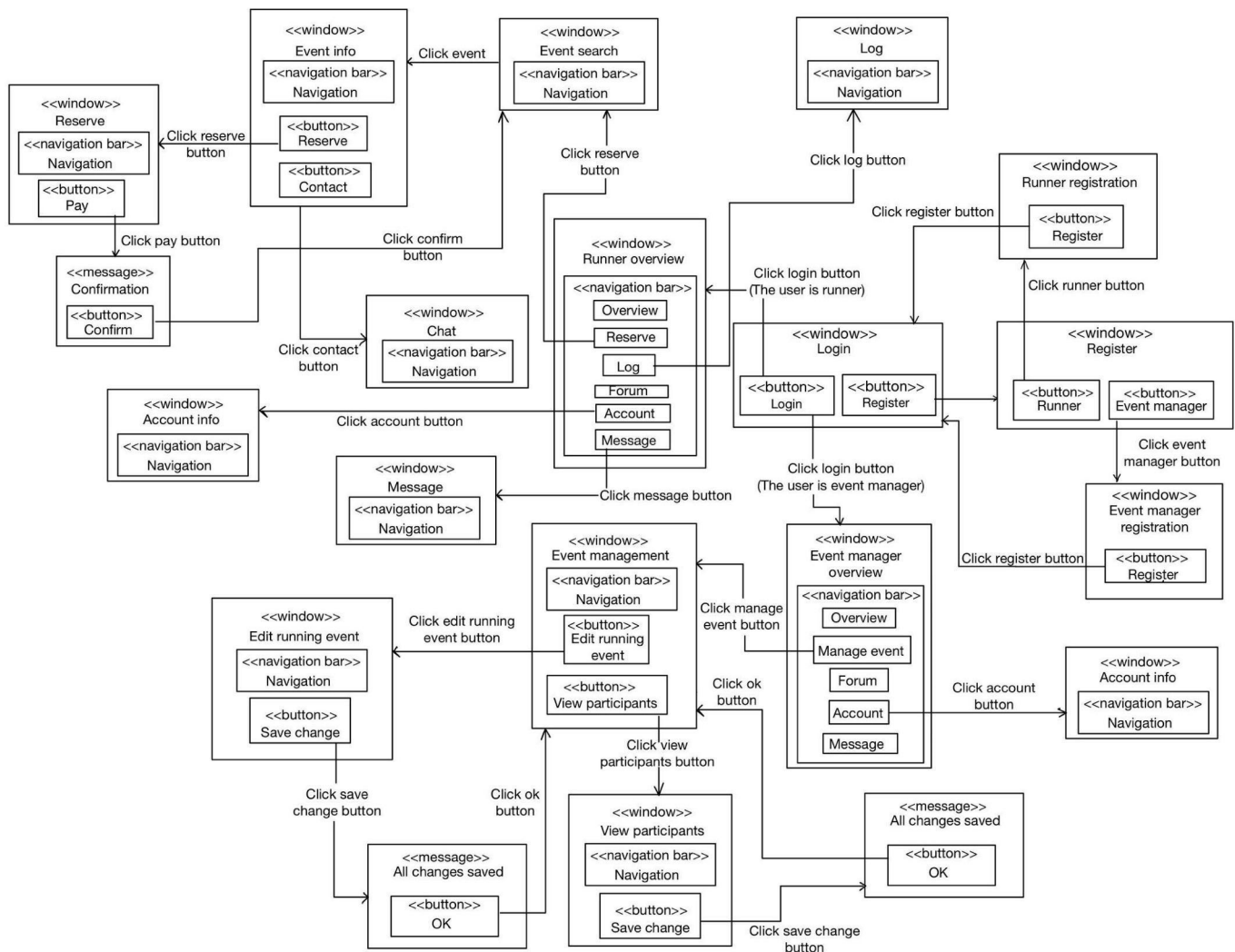


Figure 20. Window Navigation Diagram (WND)

5.Detail real use case description

Use Case Name: Reserve Running Event	ID: UC-1	Important Level: High
Primary Actor: Runner	Use Case Type: Real, Essential	
Stakeholders and Interests: 1. Runner - wants to reserve running event 2. Event manager - wants to monitor event reservations		
Brief Description: This use case support runners to reserve desired events on the website.		
Precondition: Runner logged in to the system.		
Trigger: Runner starts select for running event on the website. Type: External		
Relationships: Association: Runner, Email Distribution System Include: Browse Running Event Extend: Make Payment Generalization:		
Normal Flow of Events: 1. User fill username and password and click login 2. User fill desired running event that he/she want to join in the browse bar 3. User select desired running event from the running result 4. User fill information form required for reserving running event 4.1 If User need to pay for the reservation, do S-1 subflow		
SubFlows: S-1: Event Payment 1. Runner select credit card or debit card as a payment method. 2. Runner fill in information required for payment form. 3. Runner click Pay 4. The system redirect runner back to search event page.		
Alternate/Exceptional Flow 4a. If the user fill in invalid information, there will be an error message and the user has to fill it in again. S-1,3a. If the user fill in invalid payment form, there will be an error message and the user has to fill it again.		
Postcondition: Runner is added to approval request list for the event.		

Figure 21. Reserve Running Event Use Case Description

Use Case Name: Manage Event	ID: UC-2	Important Level: High
Primary Actor: Event Manager, System Administrator	Use Case Type: Detail, Essential	
Stakeholders and Interests: 1. Event Manager 2. System Administrator 3. Runner		
Brief Description: This use case support primary actor to manage the running event in the system.		
Precondition: The primary actor logged in to the system.		
Trigger: Primary actor start to browse for various events displayed on the webpage. Type: External		
Relationships: Association: Event Manager, System Administrator Include: Extend: Approve Participant, Make New Running Event, Edit Running Event Generalization:		
Normal Flow of Events: 1. The Event manager login into the account. 2. The Event manager browses to the manage event page. 3. The Event manager choose to perform one of these operation 3.1. If the primary actor want to approve a participant,click View participants. 3.2. If the primary actor want to create a new event, click Plus button. 3.3. If the primary actor want to edit a running event, click Edit Running Event. 4. The system display save changed event information to the primary actor.		
Postcondition: A log is created for each operation done for each event.		

Figure 22. Manage Event Use Case Description

Use Case Name: Register New Account	ID: UC-3	Important Level: High
Primary Actor: User	Use Case Type: Detail, Essential	
Stakeholders and Interests: 1. Runner 2. Event Manager 3. System Administrator		
Brief Description: This use case support user to register an account into the system.		
Precondition: The user is not registered to the website before.		
Trigger: The user want to use the system and decided to register a new account. Type: External		
Relationships: Association: User Include: Extend: Register Runner Account, Register Event Manager Generalization:		
Normal Flow of Events: 1. In the login page, User click register button. 2. User fill in desired username and password. 3. User choose to click between register as runners and register as event manager. 3.1 If user click register as runners fill in firstname, lastname,gender, ID number,address,phone number,email textboxes. 3.2 If user click register as event manager fill in firstname, lastname,gender, ID number,address,phone number, email and bank account information textboxes. 4. User clicks register		
Alternate/Exceptional Flows: 2a. If username is duplicated in the system, it will pop an alert message telling the user to fill in new username		
Postcondition: An account is created on the system.		

Figure 23. Register New Account

6. Capture of screens of the UI design prototype

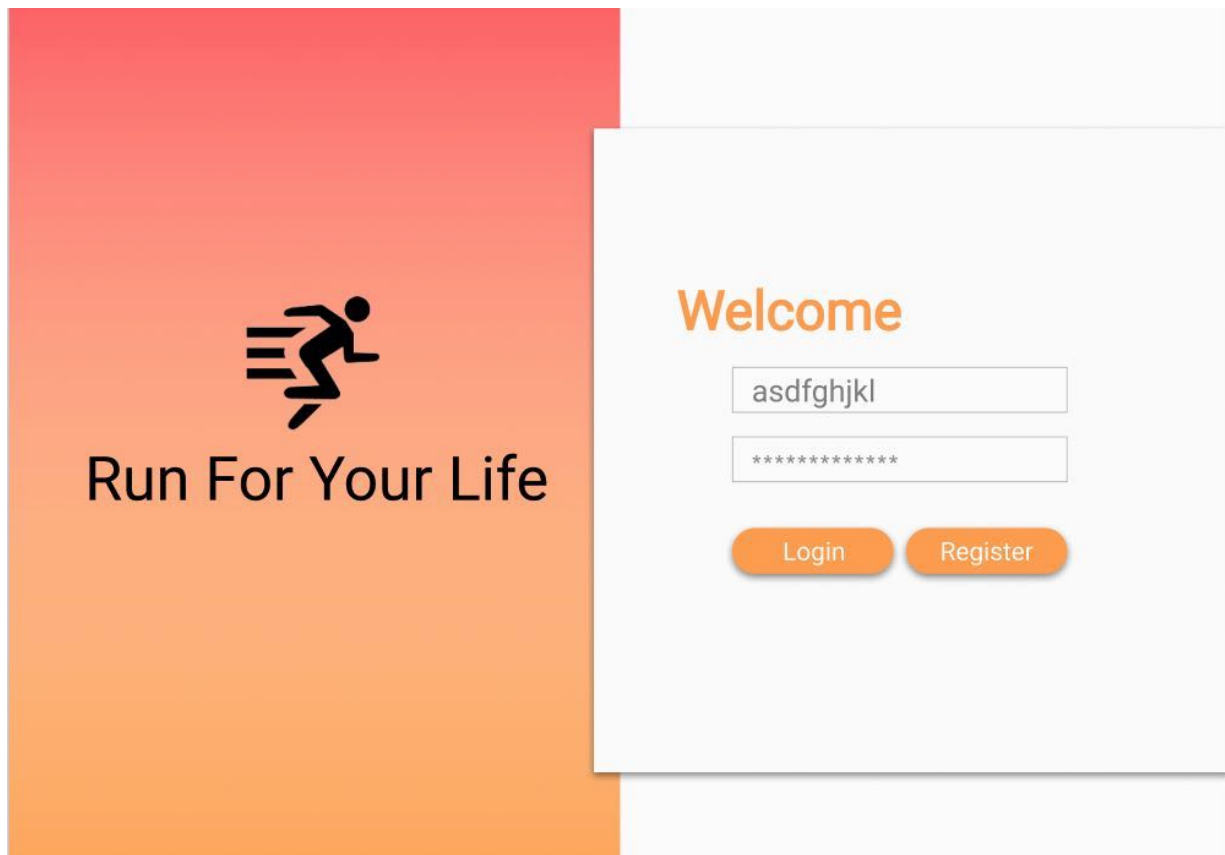
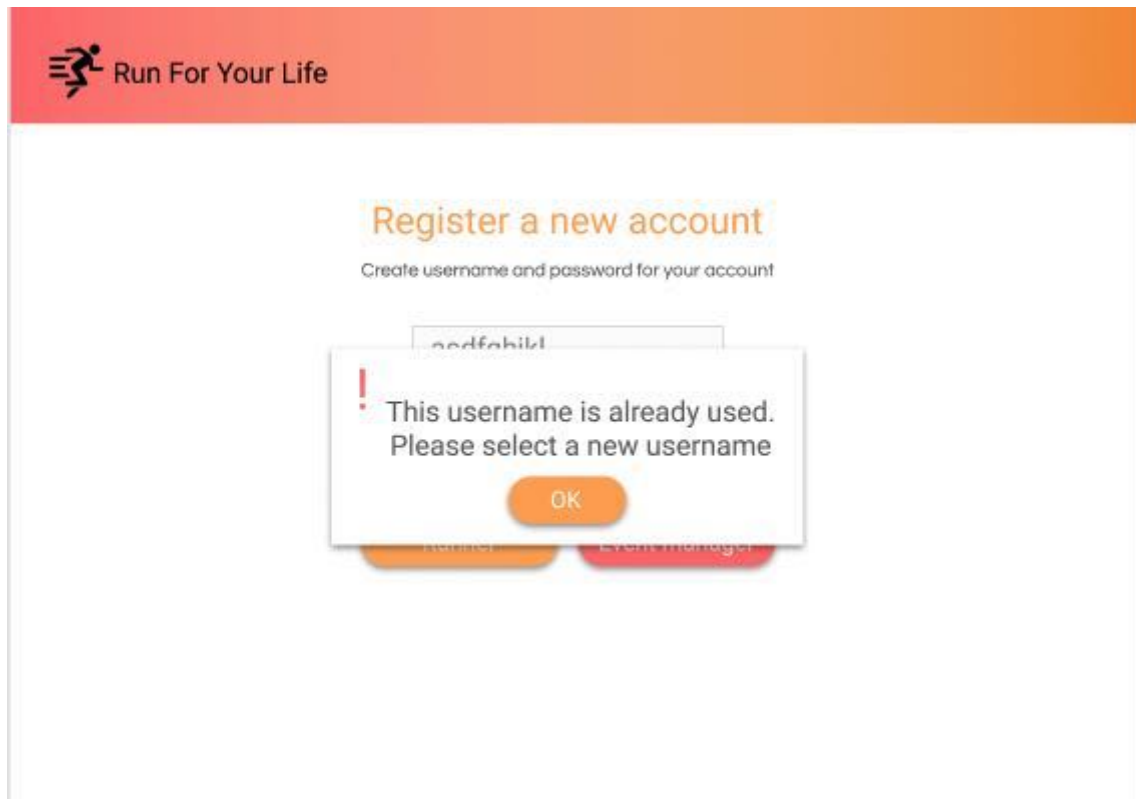


Figure 24. Login page for all users

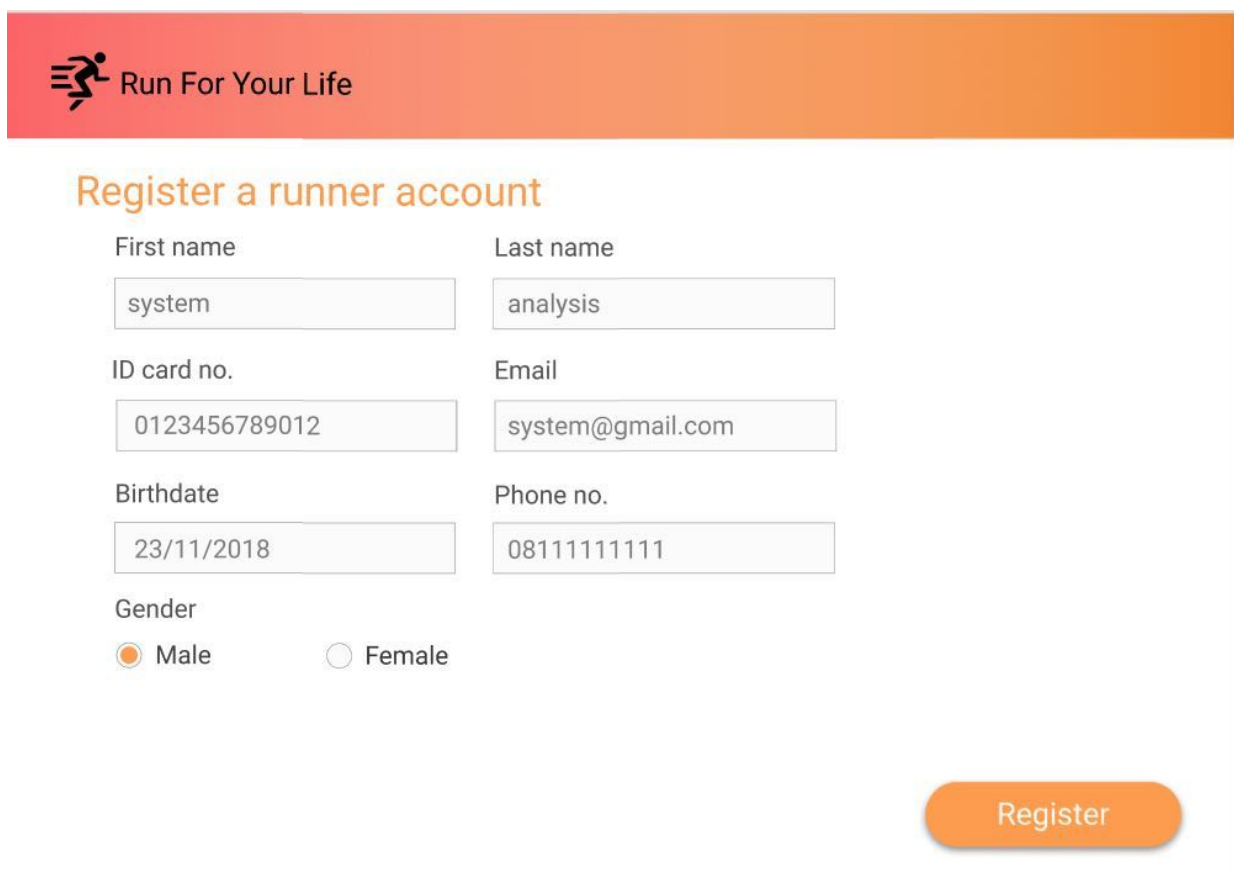


Figure 25. Register new account




The screenshot shows the 'Run For Your Life' registration page. At the top, there is an orange header with the logo and text 'Run For Your Life'. Below the header, the main content area has a title 'Register a new account' and a subtitle 'Create username and password for your account'. A text input field for the username contains the text 'sodfahib1'. A modal alert box is displayed in the center, with a red exclamation mark icon and the text: 'This username is already used. Please select a new username'. Below the text is an orange 'OK' button. At the bottom of the page, there are two buttons: 'Runner' and 'Event manager'.

Figure 26. Alert message from duplicated username entered



The screenshot shows the 'Run For Your Life' runner registration page. At the top, there is an orange header with the logo and text 'Run For Your Life'. Below the header, the main content area has a title 'Register a runner account'. The form consists of several input fields arranged in two columns. The first column contains: 'First name' with the value 'system', 'ID card no.' with the value '0123456789012', 'Birthdate' with the value '23/11/2018', and 'Gender' with radio buttons for 'Male' (selected) and 'Female'. The second column contains: 'Last name' with the value 'analysis', 'Email' with the value 'system@gmail.com', and 'Phone no.' with the value '08111111111'. At the bottom right of the form is an orange 'Register' button.

Figure 27. Runner registration page

 Run For Your Life

Register an event manager account

First name

Last name

ID card no.

Email

Birthdate

Phone no.


Gender



☒ Male
 ☐ Female

Bank account information

Register

Figure 28. Event manager registration page

 Run For Your Life

[Overview](#)
[Reserve](#)
[Log](#)
[Forum](#)



Event calendar

November 2561

Sun	Mon	Tue	Wed	Thu	Fri	Sat
28	29	30	31	1 Nov	2	3
Running event 4 at Chiangmai	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	Running event 30 at Praram9	1 Dec
2	3	4	5	6	7	8

Figure 29. Overview for runner account

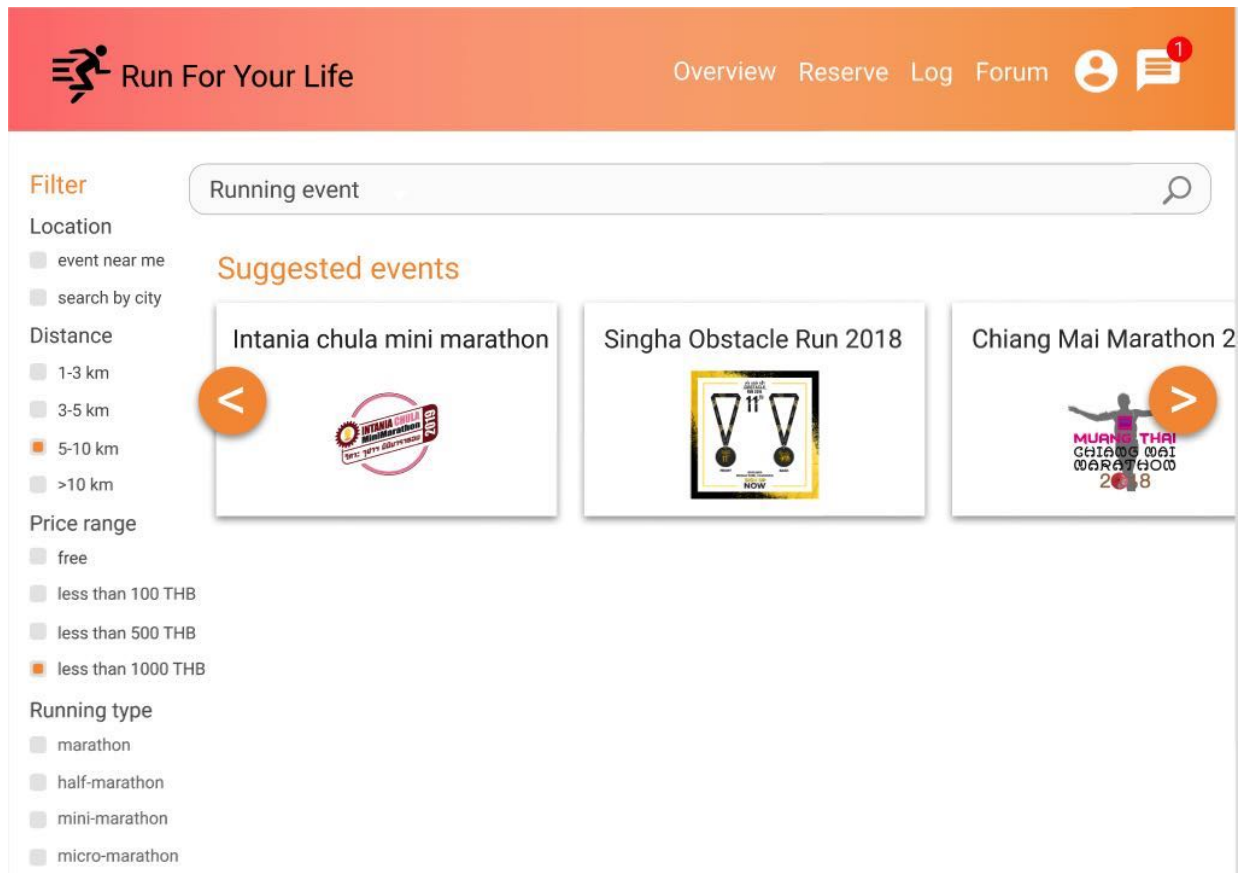


Figure 30. Running event search page

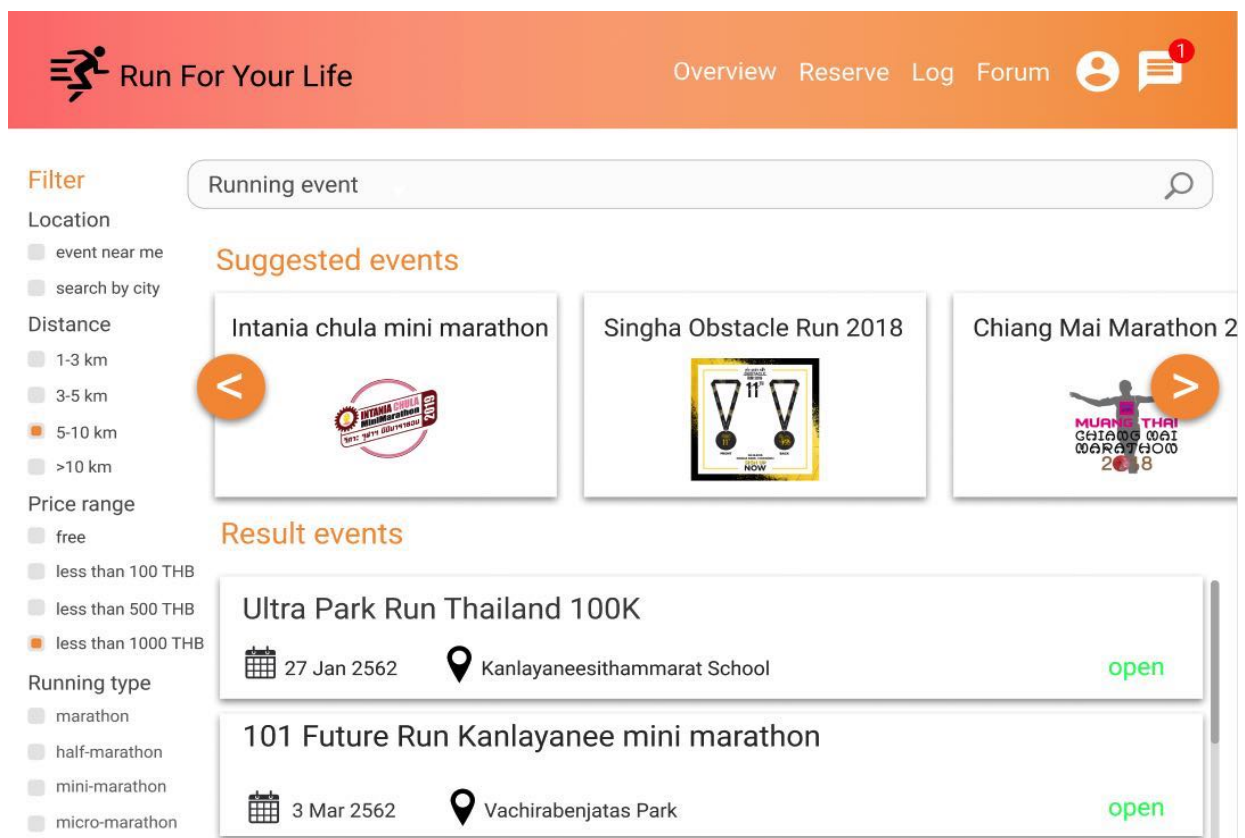







Figure 31. Running event search result



 Run For Your Life


Overview Reserve Log Forum  

Event information

Intania chula mini marathon

 13/01/2018  Faculty of engineering , Chulalongkorn university



Intania chula mini marathon


Price: 550 THB


Distance: mini marathon 10 km
fun run 5 km


Register date: 27/10/2018 11.11 AM

...

Comments





 Good event





 Recommended!

Figure 32. Running event information page

 Run For Your Life

Overview Reserve Log Forum  

Reserve running event

Runner information

Please fill in the required information.

Shirt size

Payment information

This running event require payment

Payment method



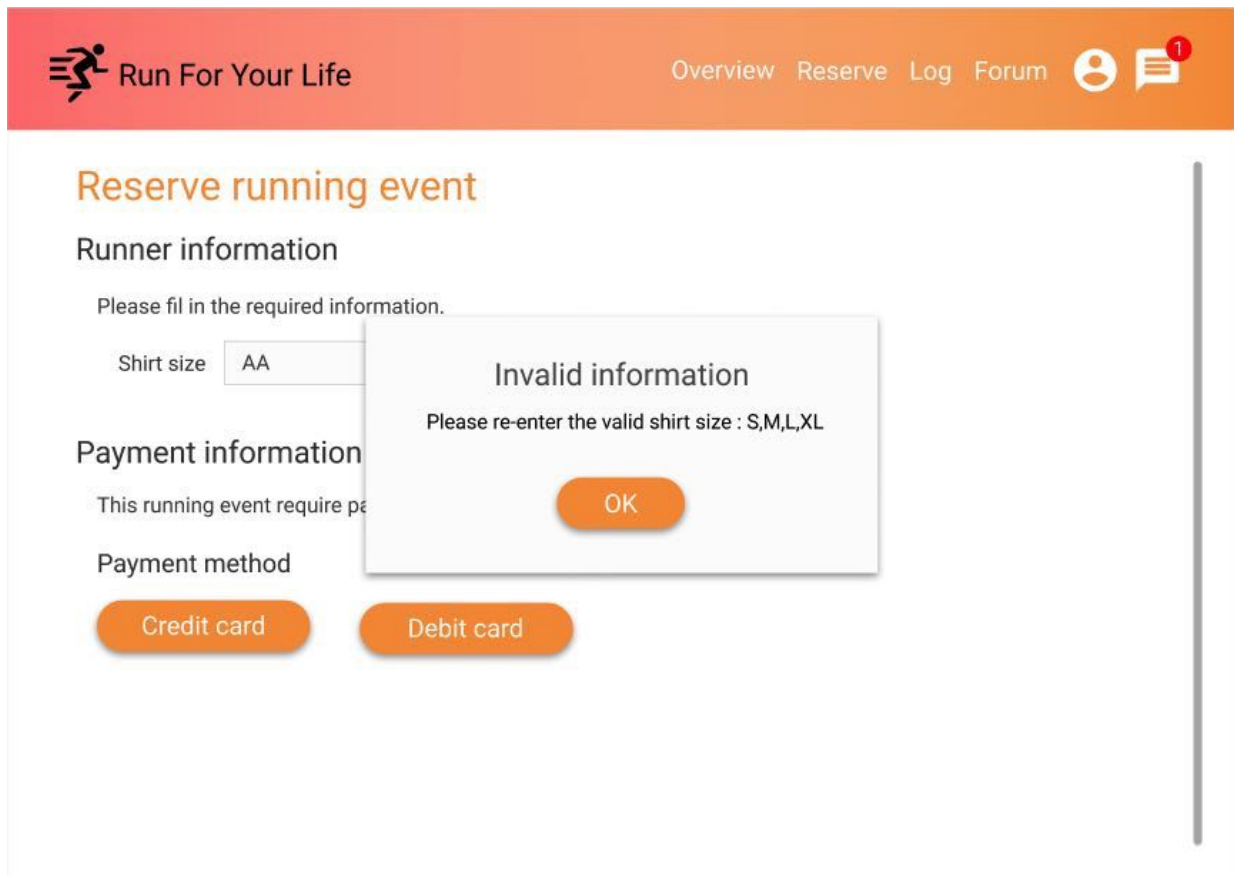
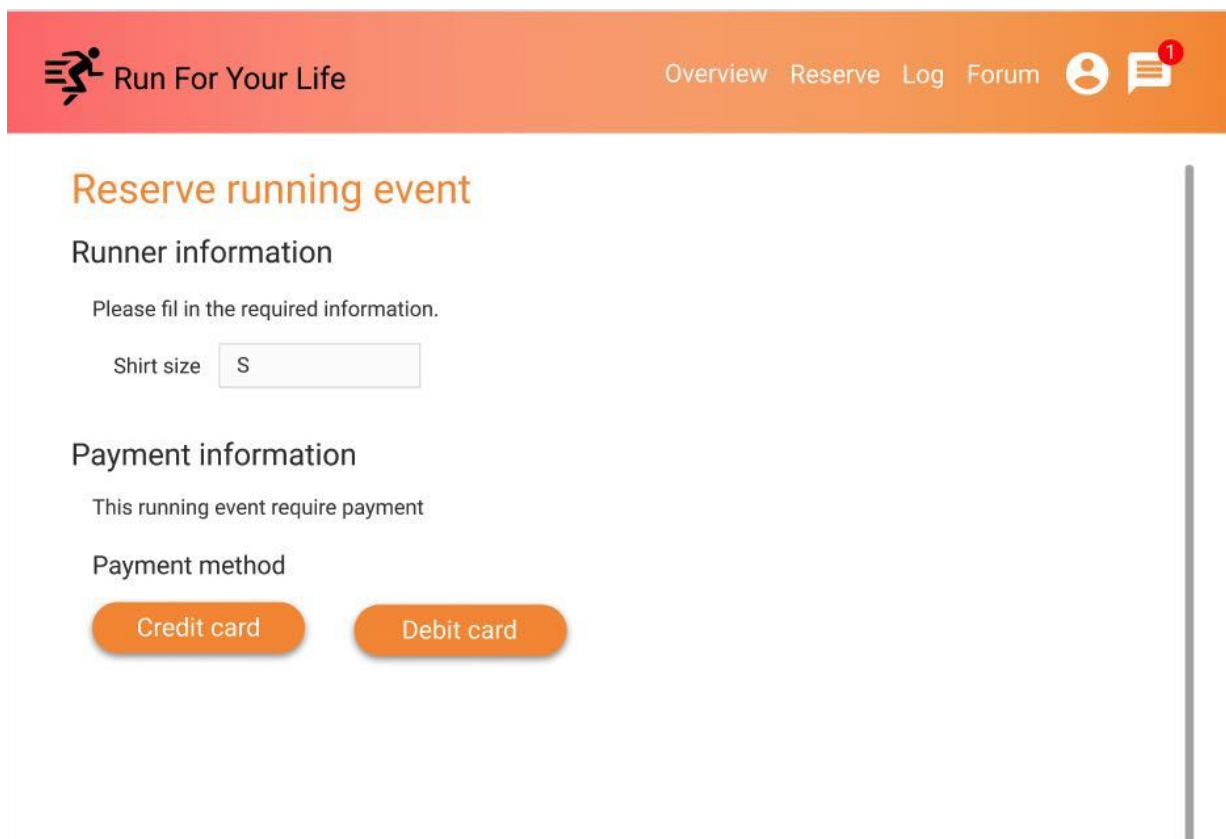


Figure 33. Reserve running event page



The screenshot shows the 'Reserve running event' page with a red header bar. The header contains the 'Run For Your Life' logo and navigation links: Overview, Reserve, Log, Forum, and a user profile icon with a notification badge. The main content area is titled 'Reserve running event' and includes sections for 'Runner information' and 'Payment information'. In the 'Runner information' section, the 'Shirt size' dropdown menu is open, showing 'AA' as the selected option. An alert dialog box is displayed over the form, titled 'Invalid information', with the message 'Please re-enter the valid shirt size : S,M,L,XL' and an 'OK' button. The 'Payment information' section includes a note 'This running event require payment' and two buttons for 'Credit card' and 'Debit card'.

Figure 34. Alert message from invalid information entered



The screenshot shows the 'Reserve running event' page with a red header bar. The header contains the 'Run For Your Life' logo and navigation links: Overview, Reserve, Log, Forum, and a user profile icon with a notification badge. The main content area is titled 'Reserve running event' and includes sections for 'Runner information' and 'Payment information'. In the 'Runner information' section, the 'Shirt size' dropdown menu is open, showing 'S' as the selected option. The 'Payment information' section includes a note 'This running event require payment' and two buttons for 'Credit card' and 'Debit card'.

Figure 35. Valid information for reserve running event page

The screenshot shows the 'Payment method' form in the 'Run For Your Life' application. The header is orange with the logo and navigation links: Overview, Reserve, Log, Forum, and user icons. The form has two tabs: 'Credit card' (selected) and 'Debit card'. Under 'Credit card', there are logos for Mastercard and VISA. The 'Card no.' field contains 'XXXX-XXXX-XXXX-XXXX' with a lock icon. The 'Expiry date' fields contain '01' and '20'. The 'Security no.' field contains 'XXX' with a card icon. A 'Pay' button is at the bottom.

Figure 36. Payment method for reserve running event

This screenshot shows the same 'Payment method' form as Figure 36, but with an alert message overlay. The alert is a white box with a grey border containing the text 'Payment information invalid' and 'Please re-enter your payment information'. An 'OK' button is at the bottom of the alert. The form elements in the background are partially visible.

Figure 37. Alert message from invalid payment information entered

The screenshot shows the 'Run For Your Life' application interface. At the top, there is an orange header bar with the app logo and name on the left, and navigation links 'Overview', 'Reserve', 'Log', and 'Forum' on the right. On the far right of the header are icons for a user profile and a message notification (indicated by a red '1'). Below the header, the main content area displays the 'Payment method' section with two buttons: 'Credit card' and 'Debit card'. Under 'Credit card', there are logos for Mastercard and VISA. Below these are input fields for 'Card no.' (containing 'XXXX-XXXX-XXXX-XXXX') and 'Security no.' (containing 'XXX'). A 'Pay' button is located at the bottom left of this section. A white confirmation pop-up is centered on the screen. The pop-up has a title 'Confirmation' and a message 'Pay 550 THB to Intania chula mini marathon'. At the bottom of the pop-up are two buttons: 'Confirm' and 'Cancel'.

Figure 38. Confirmation pop-up for payment

This screenshot shows the same 'Run For Your Life' application interface as Figure 38. The 'Payment method' section and the 'Pay' button are visible in the background. A white success message pop-up is centered on the screen. The pop-up has a title 'Reservation succeeded' and a message 'The system will redirect to overview page'. At the bottom of the pop-up is a single button labeled 'OK'.

Figure 39. Reservation succeeded message

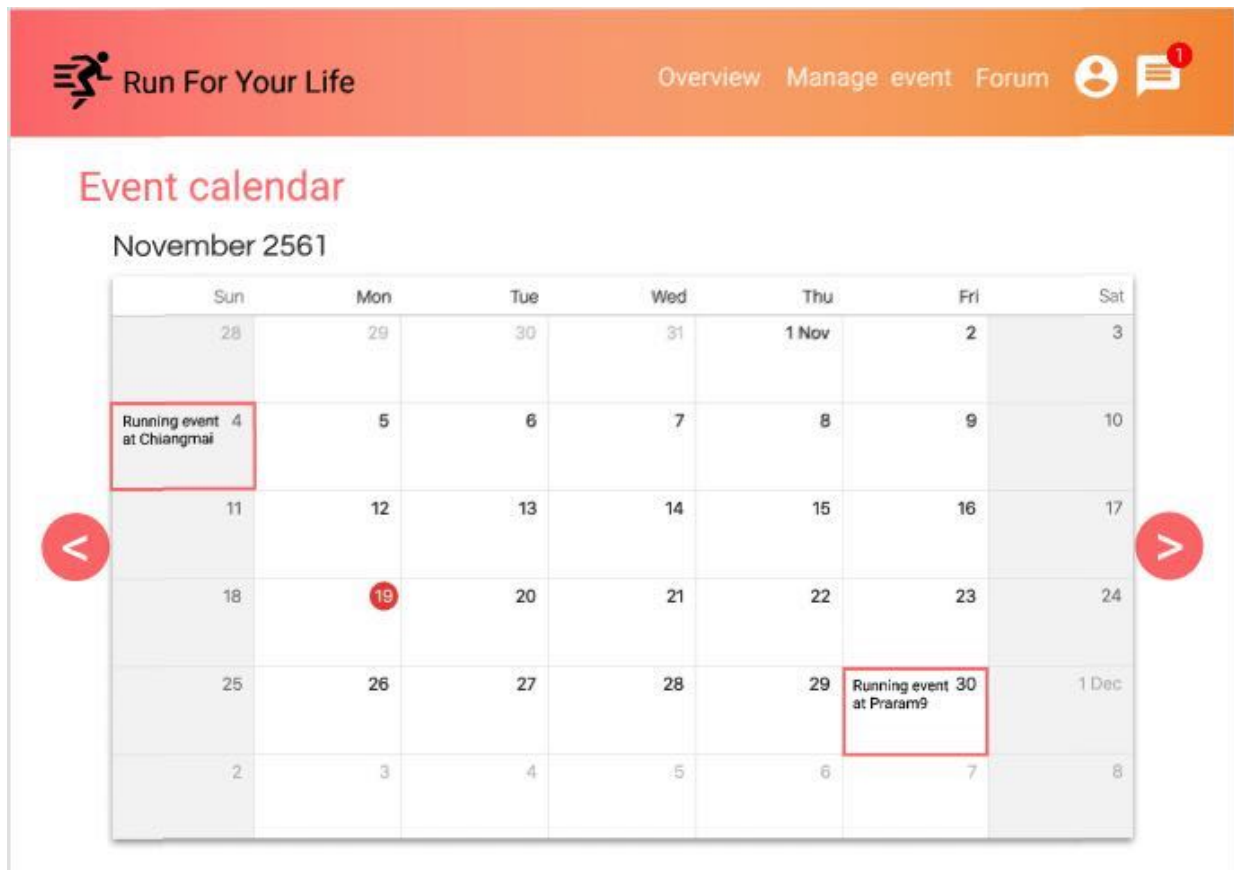


Figure 40. Overview for event manager account

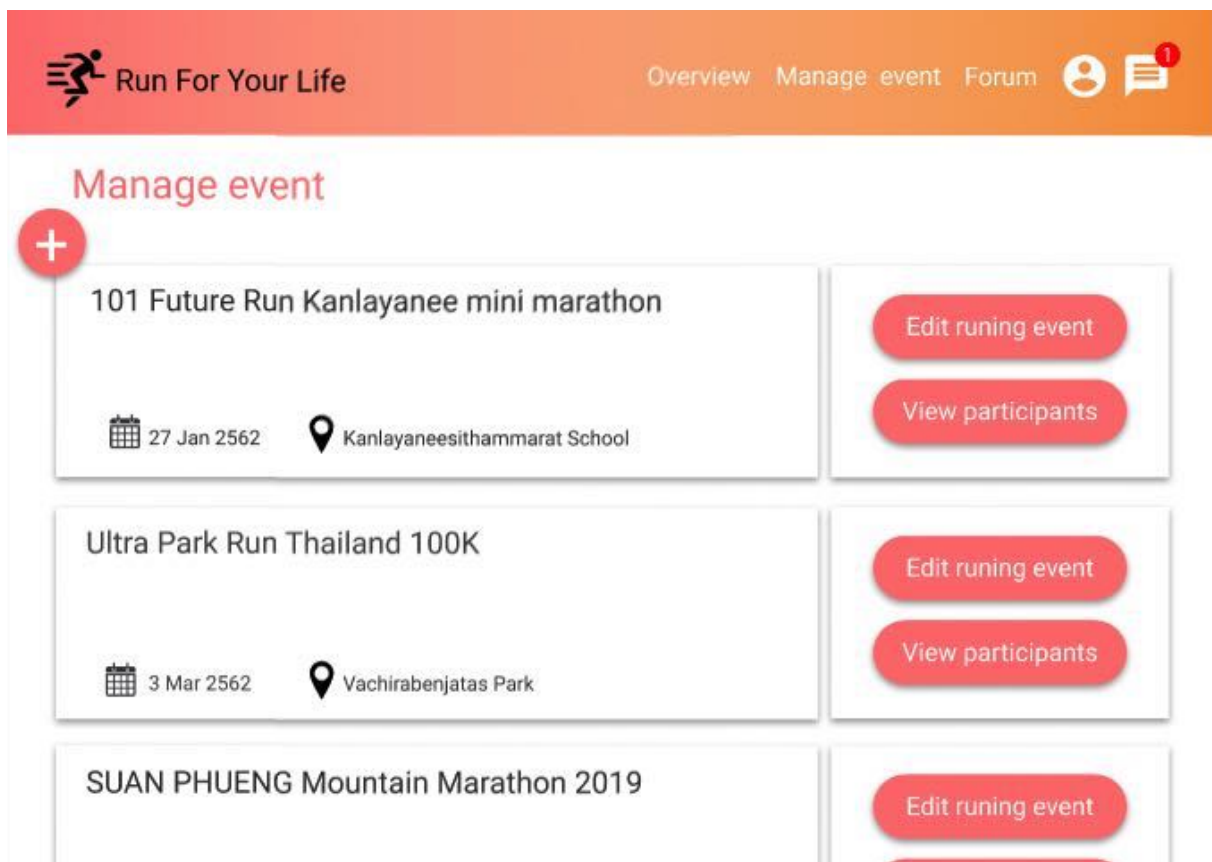





Figure 41. Manage event page


Run For Your Life


Manage event
Forum





Participants

	Participant	Reserved date	Shirt size	Status	
1	System analysis	20/11/2018	S	Active	<button>Remove</button>
2	Operation system	20/11/2018	M	Active	<button>Remove</button>
3	Data base	20/11/2018	L	Cancelled	<button>Remove</button>
4	Programming language	20/11/2018	S	Waiting	<button>Approve</button>
5	Embedded system	20/11/2018	XL	Waiting	<button>Approve</button>

Save change

Figure 42. View participants page


Run For Your Life

Overview
Manage event
Forum



Participants


	Participant	Reserved date	Shirt size	Status	
1	System analysis	20/11/2018	S	Active	<button>Remove</button>
2	Operation system			Active	<button>Remove</button>
3	Data base			Cancelled	<button>Remove</button>
4	Programming lan			Waiting	<button>Approve</button>
5	Embedded system			Waiting	<button>Approve</button>



Save change

All changes saved

OK



Figure 43. All changes saved information message


 Run For Your Life


OverviewManage eventForum

Edit running event

Ultra Park Run Thailand 100K


 3 Mar 2562 Vachirabenjatas Park



Event information

Ultra park run thailand 100K at Rot fai park

Save change



Figure 44. Edit running event page


 Run For Your Life


OverviewManage eventForum

Edit running event

Ultra Park Run Thailand 100K

 3 Mar 2562 Vachirabenjatas Park

Event information

Ultra park run thailan

All changes saved
OK

Save change

Figure 45. All changes saved information message

10. Physical architecture design

Architectural model

Since our system is web application platform, we can separate our physical system into 3 main parts which are client for users, server for application main logics and database management. This Architecture is considered as a 3-tier physical architecture.

We decided to implement these architecture on cloud by using PC and mobile devices as a client for HCI layer, Amazon AWS EC2 for problem domain layer and Amazon AWS RDS for database management layer.

Green IT

Green IT that we choose in this project is energy consumption of data centers and desktops. Our data will center in cold places and will use alternative energy. Also, we use cloud virtualization to reduce number of servers. Also, our project encourage paperless office. It reduce a lot of paper use deploying running event managing system on server instead of managing it by hand.

Diagram

We present the architectural model by deployment diagram and network diagram with all of relationships between all hardware devices and all software instances of the system.

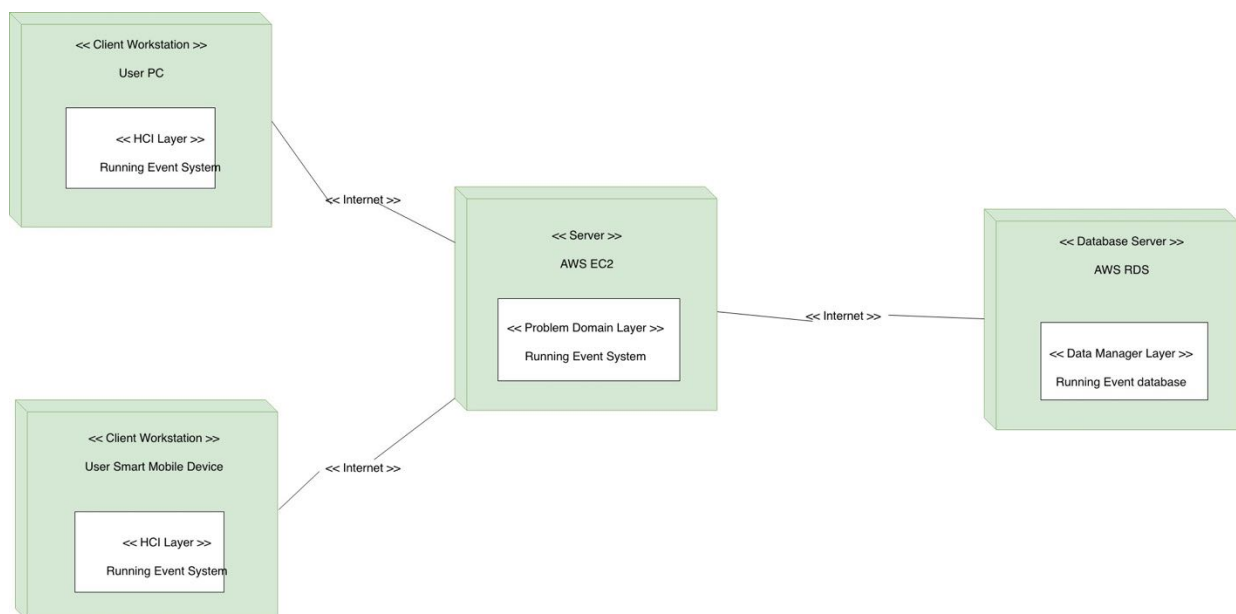


Figure 46. Deployment diagram

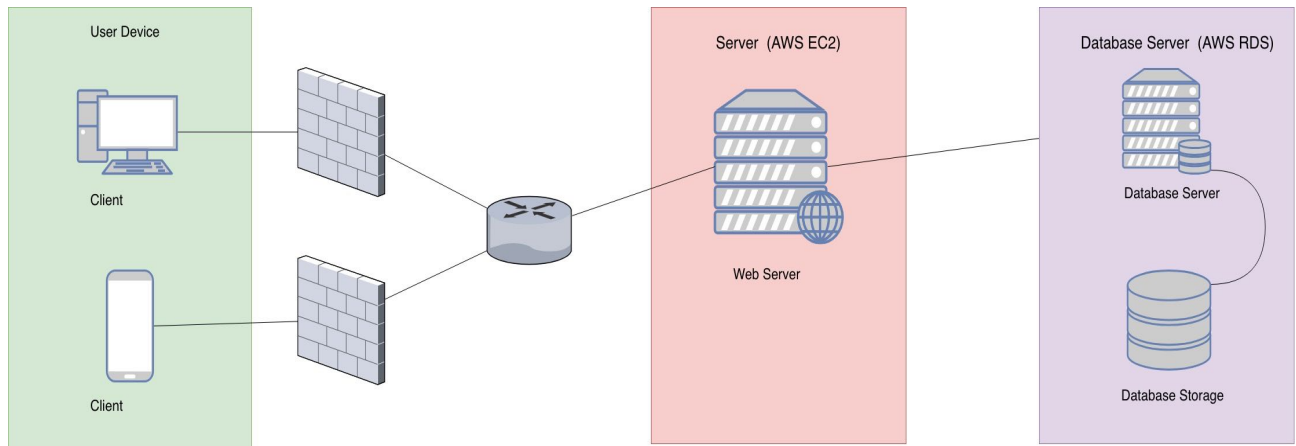


Figure 47. Network diagram

Nonfunctional requirements and physical architecture design

No.	Type of Non-Functional Requirement	Requirement	Effect for designing architectural model
1.	Operational Requirements	The system shall operate in every mainstream browser.	Design UI and method to operate on according browser
		The system shall make automatic backup of its database daily	Implement the function to make the snapshot of the database during the low time of the day.
2.	Performance Requirements	The system shall have the response time less than 2 seconds for every interaction between system and user.	Design architectural model for handle many request according to the usage at the time by resize the cloud instance and storage to appropriate size.
3.	Security Requirements	The system shall allow only event managers to edit his or her running event.	Design class and method to classify user according to user role of the system
		The system shall allow only system administrator to view the full participation log and transaction log of the system.	Design class and method to classify user according to user role of the system
4.	Cultural & Political	The system shall be available in English.	Design UI for support English language

	Requirements	The system may be available in Thai	Design UI for support Thai Language
		The system shall display the running events only in Thailand.	

Table 5. Nonfunctional requirements and physical architecture design

Hardware and software specification

For the maximum performance of the system that our team target, we deeply consider at the detail of the hardware and software specification. And we aggregate the detail in the table below.

Specification	Client Workstation	Server (AWS EC2)	Database Server (AWS RDS)
Operating System	Window 10 Pro 64 bit	Ubuntu 18.04 LTS	CentOS 7
Special Software	Internet Explorer, Firefox, or Chrome (Web Browser)	<ul style="list-style-type: none"> • NginX (Web server) • ReactJS (Web Application) • KoaJs (Application) • NodeJs (Application) • Gy Docker (Container) • Kubernetes (Orchestrator) 	PostgreSQL
Hardware	Intel Core i7 6950X ASUS RAMPAGE V EDITION 10 128 GB (32 x 4) Corsair Vengeance LED RED Bus 3200MHz 2x ASUS ROG Strix GeForce® GTX 1080Ti 2TB SAMSUNG 970 PRO SSD	10 TB SSD 72 vCPU 144 GB RAM	10 TB HDD 12vCPU 10 GB RAM

	Corsair AX1500i		
Network	1Gbps Ethernet	1Gbps Ethernet	1Gbps Ethernet

Table 6. Hardware and software specification

Verifying and validating physical architecture design

1. The architectural model by deployment diagram and network diagram is consistent with the hardware and software instance.
2. The non-functional requirements are consistent with the deployment diagram and network diagram shown above.
 - a. We used three-tier layer architecture to separate the human interaction layer, the problem domain layer and the data management layer. This allows us to fulfil our cultural non-functional requirement in which our system should support both Thai and English. The separation of HCI layer makes adding more languages on the user side easier. This is reflected in the deployment diagram above.
 - b. We use AWS RDS database service for database management. This allows us to make snapshot backup of the database easily; furthermore, the database can be scaled according to the usage need during particular period.
 - c. We reflected multiple platforms support in our deployment diagram. This ensures certain delivery in both desktop and Mobile platform.

11. Impact of the design solution

The running event management system has impact of analysis and design solution in an environmental, societal, economic, and global issue as following.

1. Environmental issue

1. Since our running event system operates on cloud system, it reduces the usage of resources and therefore reduce waste and pollution.
2. Reduce the usage of papers for example, using the payment form online reduce the bill paper.

2. Society issue

1. Runners will have chance to communicate and keep in touch among others and thus creating a runner's society.
2. Reduce work of the event manager so he/she will have time more to enhance the quality of the running event.

3. economic issue

The running event system will encourage runners to participate in more running event, and as a result, increase his or her spending accordingly.

4. global issues

As the people are more aware to the advantage of running activities, people will start considering benefit to their own health. This will contribute in solving many world issues such as obesity.

Part II Installation and operation phase document

1. Conversion plan

Conversion is the technical process by which a new system replaces an old system. It consists of steps to conversion plan and conversion dimensions which is stated below.

1.1 step to a conversion plan

1. Buy & install hardware - Our running system is a web application so we don't have to install any specific hardware as we are using cloud platform for our web application server and database server.

2. Install software - Our running system is web application so users don't have to install any software. However, the developer team have to install software for work, for example, PostgreSQL, Internet explorer.

3. Convert data - Before using the new to-be system, the system administrator have to use some previous data from as-if system for example, the information for the upcoming running event.

1.2 conversion dimension

1. Conversion style - As many users still feel accustomed to the as-is system, the direct conversion (instantly replacing the old system) may not be a good choice. It is more proper to use parallel. When the number of users of our system reach sufficient level, the as-is system will closed. It is more beneficial since it is more likely to find bugs and help the developers improve the system in the next version.

2. Conversion location - Since there are too many organizers of running event, we cannot convert all location at the same time, However, using pilot location will take a lot of time so we decided to use phased location.

3. Conversion modules - We will use the whole system conversion module since our running system is web application so in order to use the full function. User have to login into our website. Also, it is more proper since many functions are linked with each other so separate to many module may cause problems and bugs.

2. Change management

While changing the system for as-is system to to-be system, we have many issues to concern in order to handle with the problems. The issues that should be considered are as following.

1. Resistant to change

Since before we use the running event managing system, the runners and event manager use social media platform before. Therefore, people who are already familiar with the system may don't want to change their usual behavior. However,

we have to encourage them and make them notice the benefit of our running event managing system. Moreover, there may be additional skill need to complete the task for staff. However, there are many ways to handle those issues for example, train the staff for those additional skill or hire new staff.

2. Revising Management Policies

Management policies provide goals, define how work processes should be performed, and determine how staff are rewarded. Since we change from as-is system to to-be system, the manager of the should revise and update their management policies to support the to-be system in order to define new work processes and rewards. There are three aspects that we should encourage the owners to use - standard operating procedures (SOPs), measurements and rewards, and resource allocation. SOPs is used in order to update how each task is performed. Measurements and rewards are needed to be revised to motivate desired behavior. The manager can allocate their resources, e.g. staff and funds, to fit with the to-be system by using resource allocation.

3. Assessing Costs & Benefits

We have to develop a list of costs & benefit from two perspectives. First is organization perspective. The organization which are event manager and the manager of the running event system can evaluate their benefit via economic feasibility in the planning phase. They can also see the cost, benefit, return on investment and break-even point which improve the assessment. Second is adopter's perspective. The adopters are the runners. They need to know what their individual costs and benefits are, i.e. what they have to pay and what will they get in return when using our system.

4. Motivating Adoption

In order to make the user know about our system and motivate them to use our system we have to use 'Informal strategy'. We use memos and presentations to convince that our system has more benefit than cost. Also, our system focuses on eliminating problems rather than providing new opportunities.

5. Enabling Adoption

In order to make sure that event manager and runners can full potentially use our system, we should train both of them as follows

- Event manager: We should use Computer based training since there are many event managers from many regions. We will create video training and documentation for them.
- Runners: We should also use Computer based training because there are too many runners to use class room training or one on one training. We will create slides and animation for them.

3. Post-implementation activities

After successful transition, Post-implementation activities are required to refreeze the organization. These activities include system support, system maintenance and project assessment.

1. System support

The project development team will provide support for the system as described below

- 1.1. The system will have online support for both runners, event manager and system administrator. The system will provide faqs and documentation for them.
- 1.2. The system administrator will be the help desk on the phone. If user has problem, they can contact the help desk and he will generate a problem report. If the problem turn out to be a bug, the help desk may pass problem report and will change request to the development team.

2. System maintenance

If the problem that help desk get from user turn out to be a bug, the help desk will generate change request to the developer to fix the bugs and update new version.

3. Project assessment

Project assessment determine what is successful and what to be improved.

1. Project Team review - the project team review will be conducted immediately after the system is installed. Goal is to repeat excellent performance and eliminate mistakes in future projects.

Project Team Review

Excellent performance

- We have meeting very frequent and people in the team work very efficiently.
- Our team understand the structure of the system very well so we nearly don't have to edit anything later.
- Our team work together so we rarely have conflict because we can ask each other immediately.
- Our team is very circumspect.

Mistakes

- Sometimes we waste too much time elaborate things.

2. System Review - The system review will be conducted several months after the system is installed. Goal is to compare estimates with actual values. It will determine whether or not the system provides the expected value. Also, it provides baseline costs for future projects.