sked-it
Analysis Model

Submitted to:

Asst. Prof. Ma. Rowena C. Solamo
Faculty Member
Department of Computer Science
College of Engineering
University of the Philippines, Diliman

Submitted by:
Dela Sierra, Joshua Joseph Riki V.
Garcia, Patric Charles
Granda, Justin Tristan Gabriel R.

In partial fulfillment of Academic Requirements
for the course
CS 191 Software Engineering I
of the
1st Semester, AY 2019-2020

Unique Reference:

The documents are stored in the https://github.com/jvdelasierra/sked-it/tree/master/02-Requirements%20Engineering/Project%20Deliverables referenced 3-sked-it-Analysis Model.pdf..

Purpose:

The purpose of this document is to formalize the Analysis Method of sked-it by deriving it from the Use Case Model and Specifications

Audience:

The target audience of this document includes the developers of the sked-it project and any person who wishes to learn more about the project.

Revision Control:

Revision Date	***Person Responsible***	***Version Number***	***Modification***
09/28/2019	Joshua V. Dela Sierra	1.0	Prepared Initial Document
10/01/2019	Joshua V. Dela Sierra	1.1	Added Boundary Classes
10/02/2019	Joshua V. Dela Sierra	1.2	Added Entity Classes

System Name: Sked-it

Description: Sked-it is an app that we are proposing to make to help people manage their own personal schedules while being able to set-up meetings with other people. The app aims to provide users with a way to make a "routine" schedule which they could update if they have other things to do. One of the main features that we will implement is a Group Scheduling system where people in the same group will be allowed to view a portion of another person's schedule (controlled by the owner of the schedule) within the same group so that the people involved in the group can easily set-up their meetings based on their common free time. A person will then be notified of their schedule with a reminder sent to their phones.

***Analysis Model*:**

Place here the class diagram of the analysis model.

Boundary Classes:

Class Name	Description
AthleteRecordUI	This is the interface of the club staff to the system whenever he or she needs to maintain athlete record. Responsibilities: public void enterAthleteData(String lastname, String firstname, String mi, String address, String postalCode, Date bday, Char gender, String status) public void enterGuardianData(String lastname String firstname, String mi, String address, String postalCode, String telephone)
MonthCalendarUI	This is the interface of the end user to the system whenever he/she needs to manage his/her monthly calendar. Responsibilities: public void addEventCalendar(boolean addOrDelete, String date, String time, String description, String groupPrivacy) public void updateEventCalendar(String eventID, String date, String time, String description, String groupPrivacy) public void deleteEventCalendar(String eventID, boolean addOrDelete)
WeeklyTemplateCalendarUI	This is the interface of the end user to the system whenever he/she needs to manage his/her weekly template calendar. Responsibilities: public void addEventCalendar(boolean addOrDelete, String date, String time, String description, String groupPrivacy) public void updateEventCalendar(String eventID, String date, String time, String description, String groupPrivacy) public void deleteEventCalendar(String eventID, boolean addOrDelete)
WeeklyCalendarUI	This is the interface of the end user to the system whenever he/she needs to manage his/her weekly calendar. Responsibilities: public void addEventCalendar(boolean addOrDelete, String date, String time, String description, String groupPrivacy) public void updateEventCalendar(String eventID, String date, String time, String description, String groupPrivacy) public void deleteEventCalendar(String eventID, boolean addOrDelete)
DailyCalendarUI	This is the interface of the end user to the system whenever he/she needs to manage his/her daily calendar. Responsibilities: public void addEventCalendar(boolean addOrDelete, String date, String time, String description, String groupPrivacy) public void updateEventCalendar(String eventID, String date, String time, String description, String groupPrivacy) public void deleteEventCalendar(String eventID, boolean addOrDelete)
LoginUI	This is the interface of the end user to the system whenever he/she needs to enter the system. Responsibility: public String loginSystem(String email, String password)
CreateProfileUI	This is the interface of the end user to the system whenever he/she needs to create

	an account for the system. Responsibility: public String createProfile(String email, String password, boolean agreeOnTerms)
ForgetPasswordUI	This is the interface of the end user to the system whenever he/she forgot the password for the system. Responsibility: public String enterDetails(String email)
LinkScheduleUI	This is the interface of the end user to the system whenever he/she needs to link a schedule to a group. Responsibility: public String enterDetails(String groupCalendarID, string eventID)
ConfirmScheduleIO	This is the interface of the end user to the system whenever he/she needs to link a schedule to a group. Responsibility: public String confirmDetails(String groupCalendarID, string eventID)

Control Classes:

Class Name	**Description**
MaintainAthleteController	This is the control that maintain athlete record. It is considered an abstract class.
AddAthleteController	This is the control that adds an athlete to the system. It extends MaintainAthleteController. Responsibilities: public void AddAthlete(Athelete a)

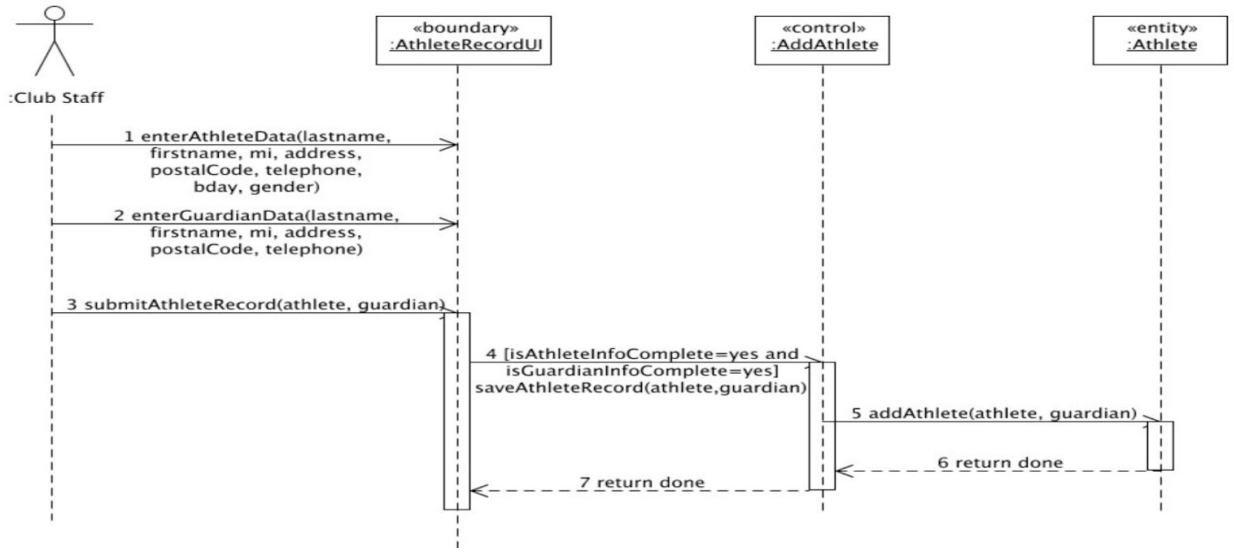
Entity Classes:

Class Name	**Description**
Athlete	This is the entity class athlete, which contains the data about the athlete. Attributes: private int athleteID private String lastname private String firstname private char gender = [M or F] … private String status; // provides the status code of the athlete eg. ACTIVE, SUSPENDED
Individual	This is the entity class Individual, which contains the data about an individual user. Attributes: private String userID private String email private String password private Schedule schedules
Schedule	This is the entity class Schedule, which contains the data about an individual user. Attributes: private String eventID private String date private String time private String description private String groupPrivacy
Group	This is the entity class Group, which contains the data about groups. Attributes: private Individual individual private String nameOfLeader

Behavioral Model:

Use-Case Name: [Name of Use Case: eg. 1.0 Applicant is assigned to a squad.]
Description: [Write the 3 to 5 sentences that describes the Use Case.]

Scenario 1: Add athlete record successfully. (Basic Flow)



Scenario 2: Athlete record with incomplete athlete data data.

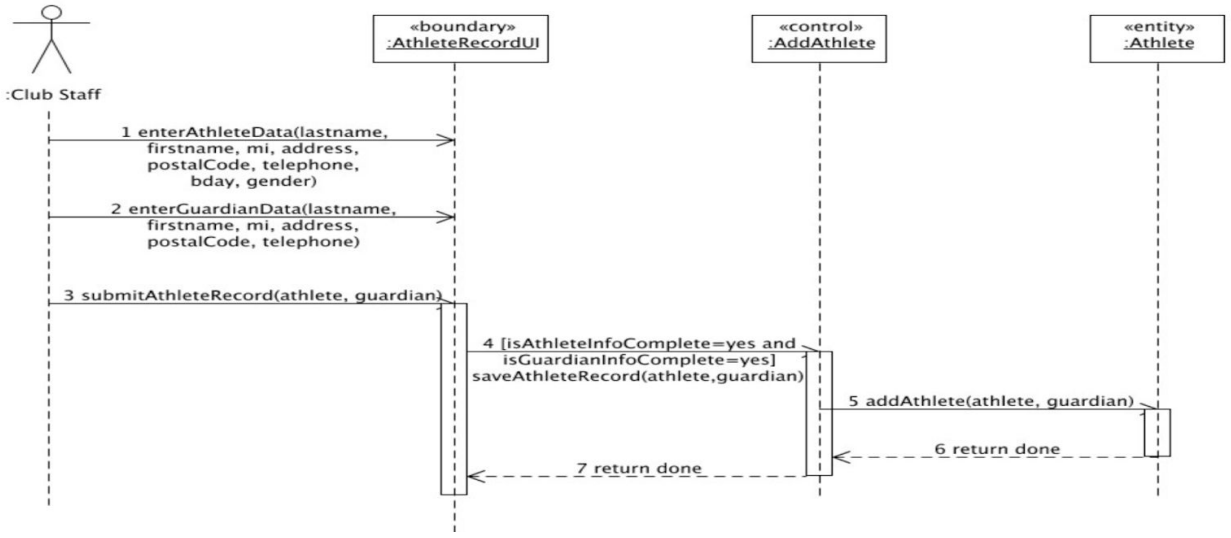
[Sequence Diagram of Scenario 2]

Scenario 3: Athelte record with incomplete guardian data.

[Sequence Diagram of Scenario 3]

Use-Case Name: [Name of Use Case: eg. 1.0 Applicant is assigned to a squad.]
Description: [Write the 3 to 5 sentences that describes the Use Case.]

Scenario 1: Basic Flow



Scenario 2:

Scenario 3: