2021

Submitted to:

Mrs. Varsha Sharda ma'am

Submitted by:

Unnati Bukhariya

EN19CS306052

Software Design with UML

Lab file

Medicaps University

Group mates:

Yash Kothane

Rohan Patidar

Omee Karve

contents

Introduction

A UML diagram is a diagram based on the UML (**Unified Modeling Language**) with the purpose of visually representing a system along with its main actors, roles, actions, artifacts or classes, in order to better understand, alter, maintain, or document information about the system

There are several types of UML diagrams and each one of them serves a different purpose regardless of whether it is being designed before the implementation or after (as part of documentation).

The two most broad categories that encompass all other types are Behavioral UML diagram and Structural UML diagram. As the name suggests, some UML diagrams try to analyze and depict the structure of a system or process, whereas other describe the behavior of the system, its actors, and its building components. .

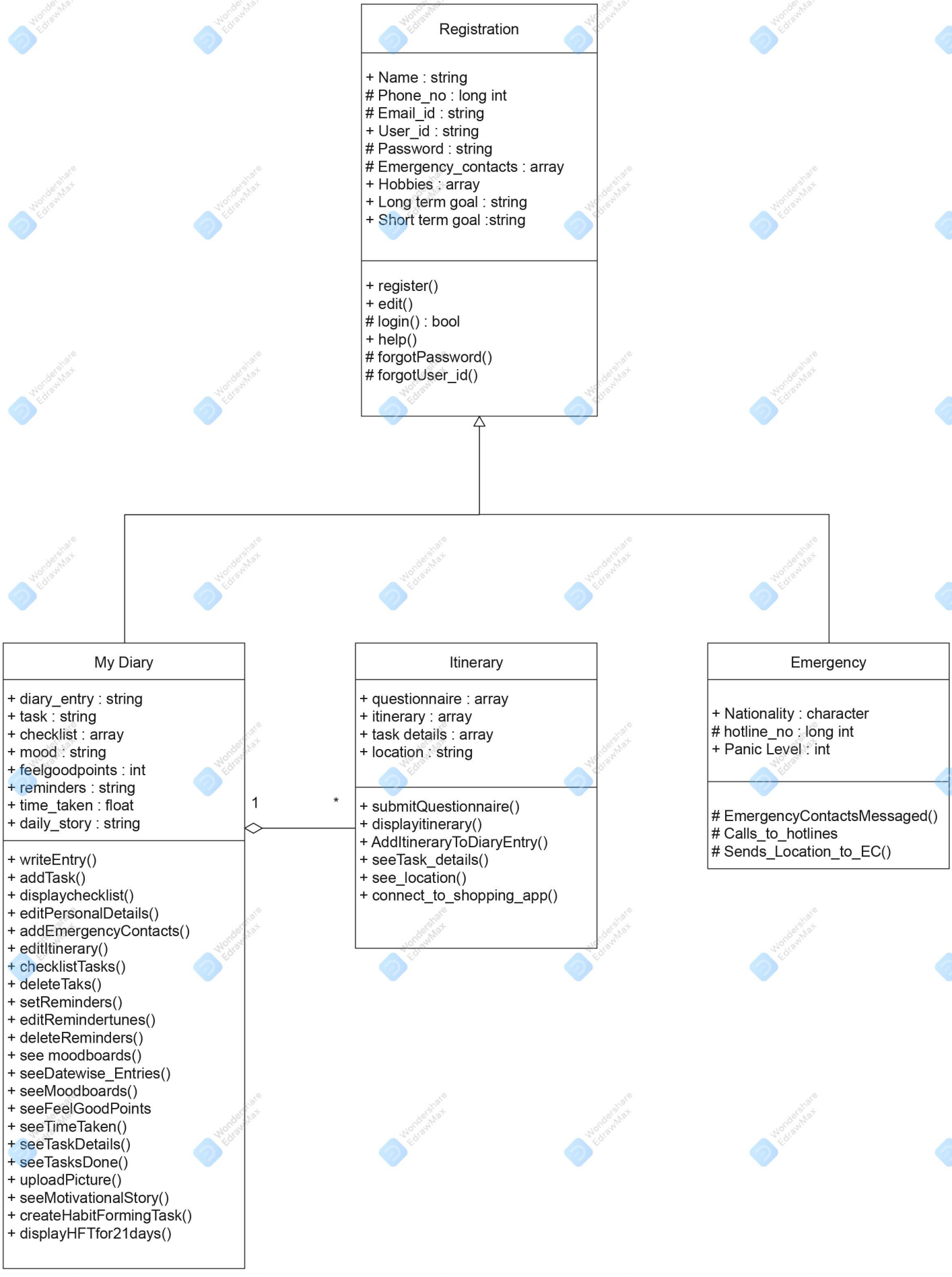
class diagram

Class UML diagram is the most common diagram type for software documentation. Since most software being created nowadays is still based on the Object-Oriented Programming paradigm, using class diagrams to document the software turns out to be a common-sense solution. This happens because OOP is based on classes and the relations between them.

In a nutshell, class diagrams contain classes, alongside with their attributes (also referred to as data fields) and their behaviors (also referred to as member functions). More specifically, each class has 3 fields: the class name at the top, the class attributes right below the name, the class operations/behaviors at the bottom. The relation between different classes (represented by a connecting line), makes up a class diagram.

Classes: User account, diary entry, itinerary and emergency.

Software used: EDraw max



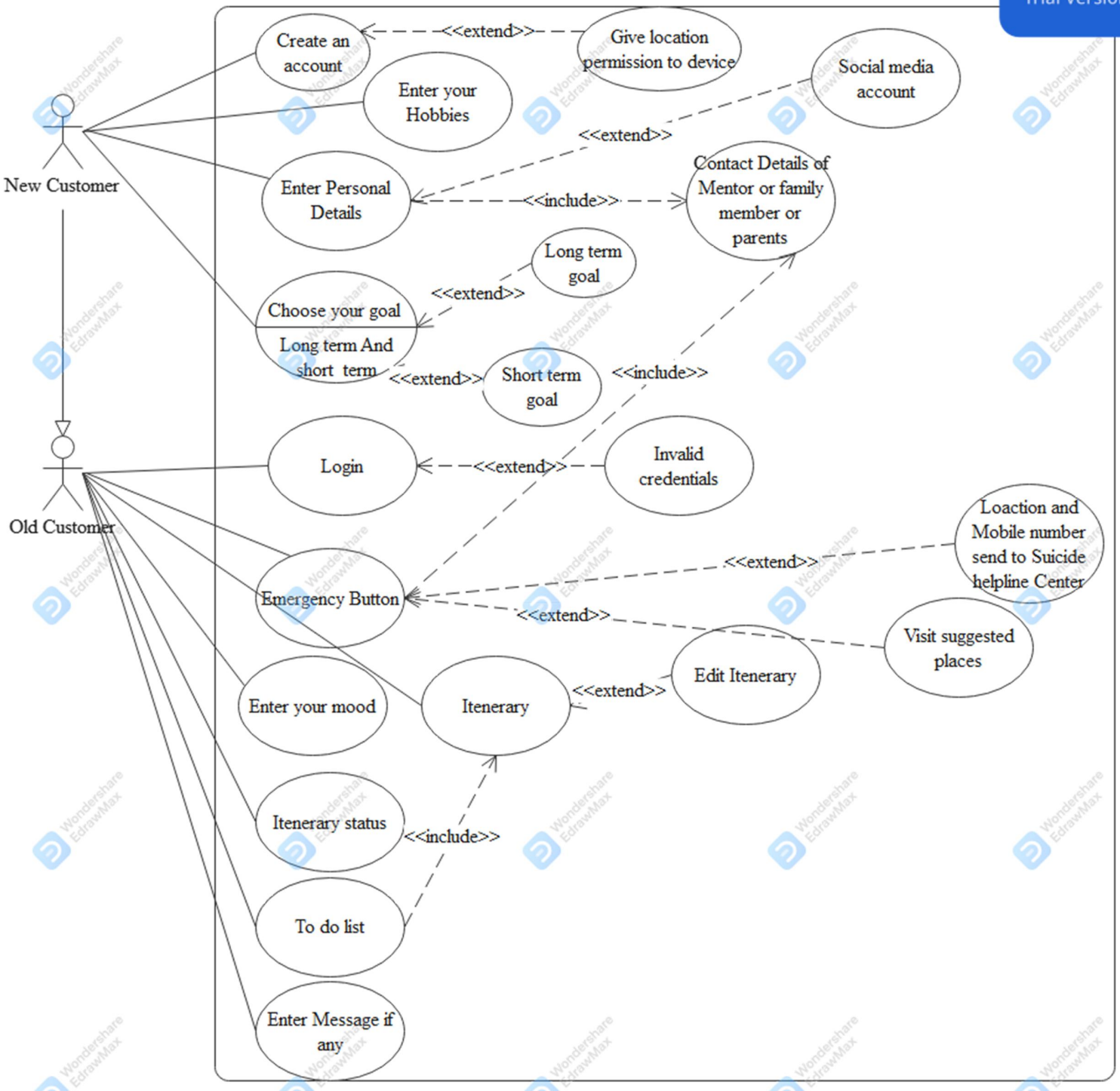
use case diagram

A cornerstone part of the system is the functional requirements that the system fulfills. Use Case diagrams are used to analyze the system's high-level requirements. These requirements are expressed through different use cases. We notice three main components of this UML diagram:

Functional requirements – represented as use cases; a verb describing an action.
Actors – they interact with the system; an actor can be a human being, an organization or an internal or external application.
Relationships between actors and use cases – represented using straight arrows.

Actors: New user, old user

Software used: EDraw max

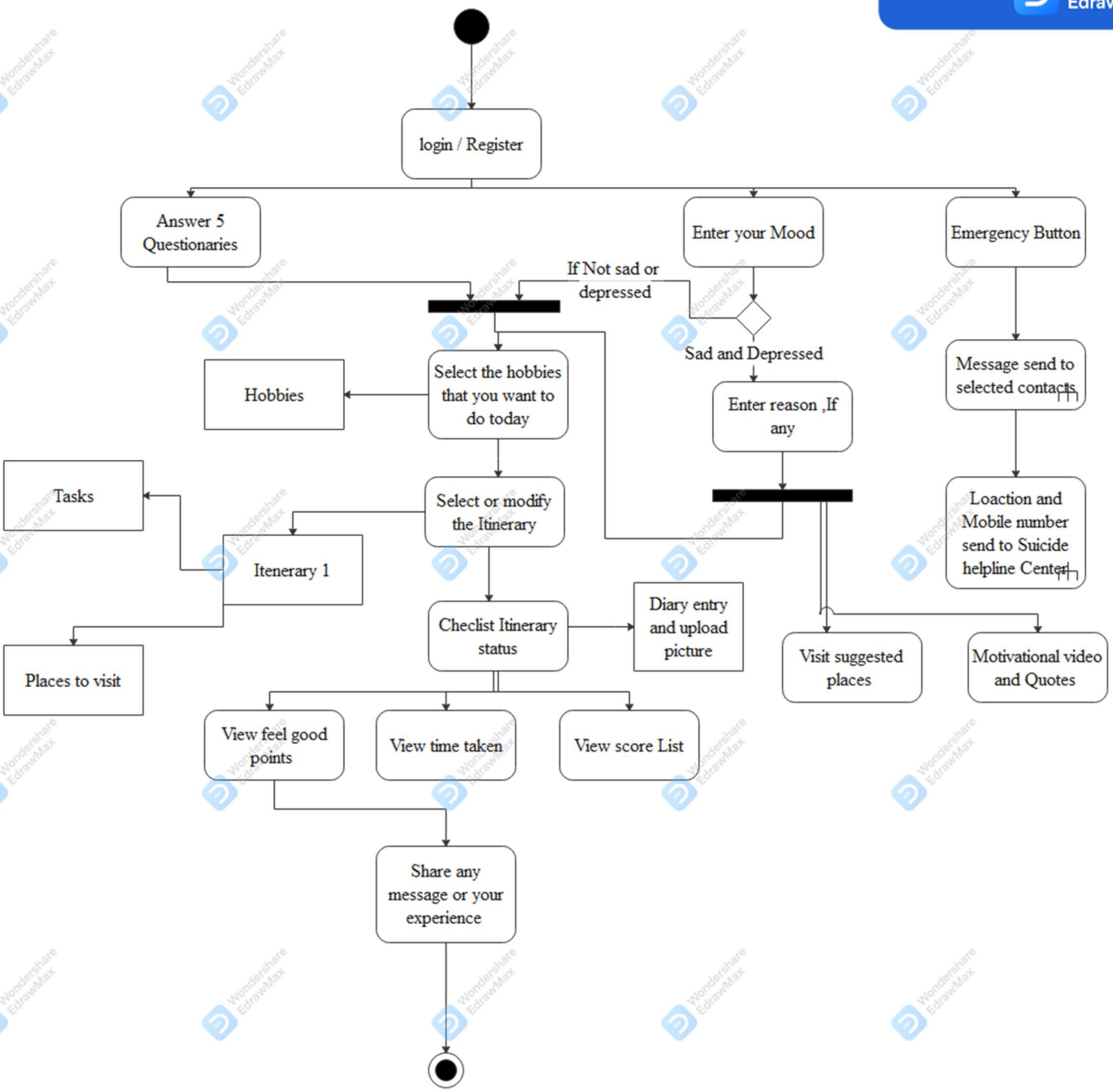


activity diagram

Activity diagrams are probably the most important UML diagrams for doing business process modeling. In software development, it is generally used to describe the flow of different activities and actions. These can be both sequential and in parallel. They describe the objects used, consumed or produced by an activity and the relationship between the different activities. All the above are essential in business process modeling.

A process is not focused on what is being produced but rather on the set of activities that lead to one the other and how they are interconnected, with a clear beginning and end. The example above depicts the set of activities that take place in a content publishing process. In a business environment, this is also referred to as business process mapping or business process modeling.

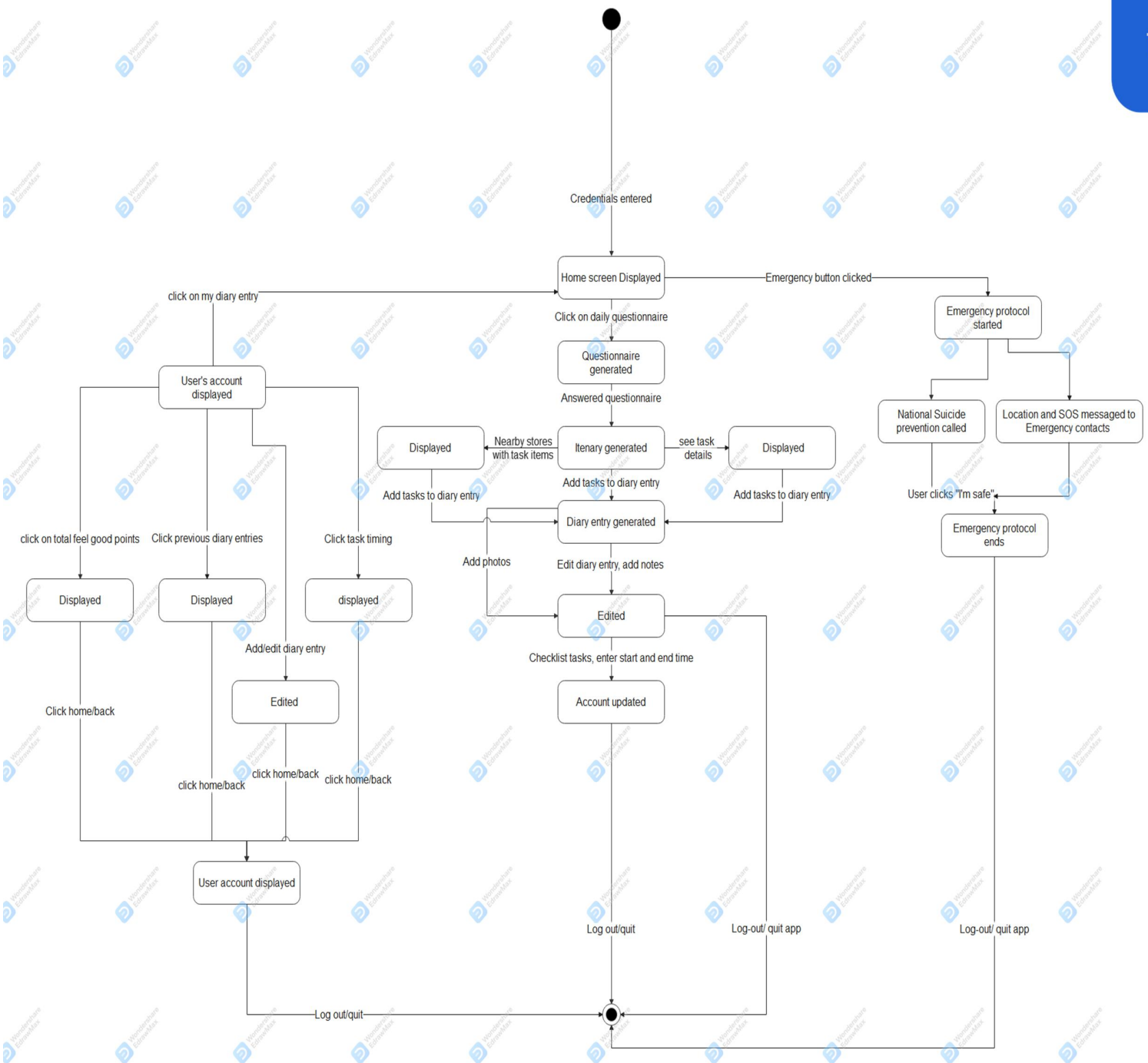
Software used: EDraw max



statechart diagram

State machine UML diagrams, also referred to as Statechart diagrams, are used to describe the different states of a component within a system. It takes the name state machine because the diagram is essentially a machine that describes the several states of an object and how it changes based on internal and external events.

Software used: EDraw max


Credentials entered
Home screen Displayed
Emergency button clicked
click on my diary entry
Click on daily questionnaire
Emergency protocol started
Questionnaire generated
User's account displayed
Answered questionnaire
National Suicide prevention called
Location and SOS messaged to Emergency contacts
Displayed
Nearby stores with task items
Itenary generated
see task details
Displayed
Add tasks to diary entry
Add tasks to diary entry
Add tasks to diary entry
User clicks "I'm safe"
Diary entry generated
Emergency protocol ends
click on total feel good points
Click previous diary entries
Click task timing
Add photos
Edit diary entry, add notes
Displayed
Displayed
displayed
Edited
Add/edit diary entry
Checklist tasks, enter start and end time
Edited
Account updated
Click home/back
click home/back
click home/back
click home/back
User account displayed
Log out/quit
Log-out/ quit app
Log-out/ quit app
Log out/quit

sequence diagram

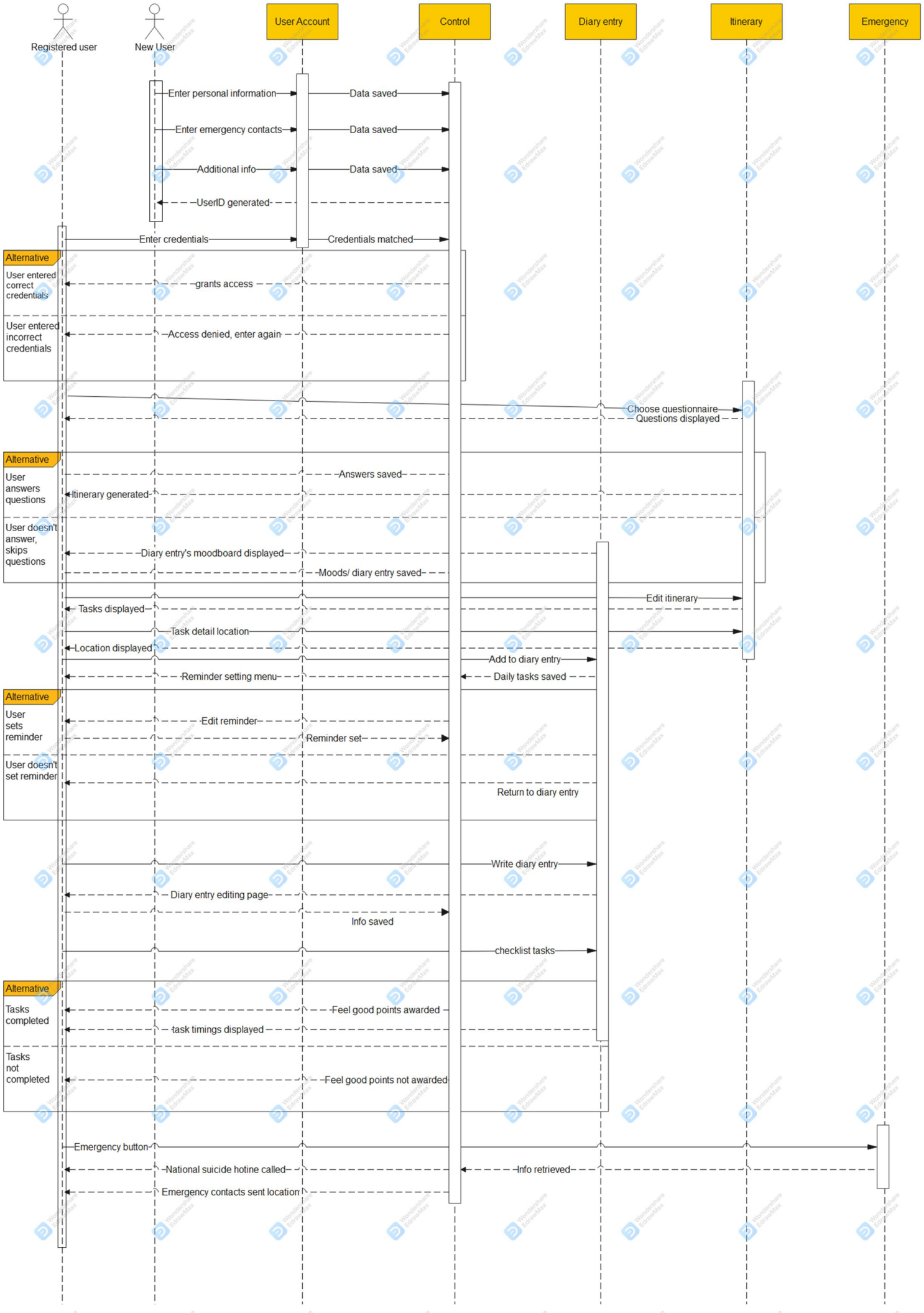
Sequence diagrams are probably the most important UML diagrams among not only the computer science community but also as design-level models for business application development. Lately, they have become popular in depicting business processes, because of their visually self-explanatory nature.

As the name suggests, sequence diagrams describe the sequence of messages and interactions that happen between actors and objects. Actors or objects can be active only when needed or when another object wants to communicate with them. All communication is represented in a chronological manner.

Objects: User Account, control, dairy entry, itinerary, emergency

Actors: New user, Registered user

Software used: EDraw max


Registered user
New User
User Account
Control
Diary entry
Itinerary
Emergency
Enter personal information
Data saved
Enter emergency contacts
Data saved
Additional info
Data saved
UserID generated
Enter credentials
Credentials matched
Alternative
User entered correct credentials
grants access
User entered incorrect credentials
Access denied, enter again
Choose questionnaire
Questions displayed
Alternative
User answers questions
Answers saved
Itinerary generated
User doesn't answer, skips questions
Diary entry's moodboard displayed
Moods/ diary entry saved
Edit itinerary
Tasks displayed
Task detail location
Location displayed
Add to diary entry
Reminder setting menu
Daily tasks saved
Alternative
User sets reminder
Edit reminder
Reminder set
User doesn't set reminder
Return to diary entry
Write diary entry
Diary entry editing page
Info saved
checklist tasks
Alternative
Tasks completed
Feel good points awarded
task timings displayed
Tasks not completed
Feel good points not awarded
Emergency button
National suicide hotine called
Info retrieved
Emergency contacts sent location

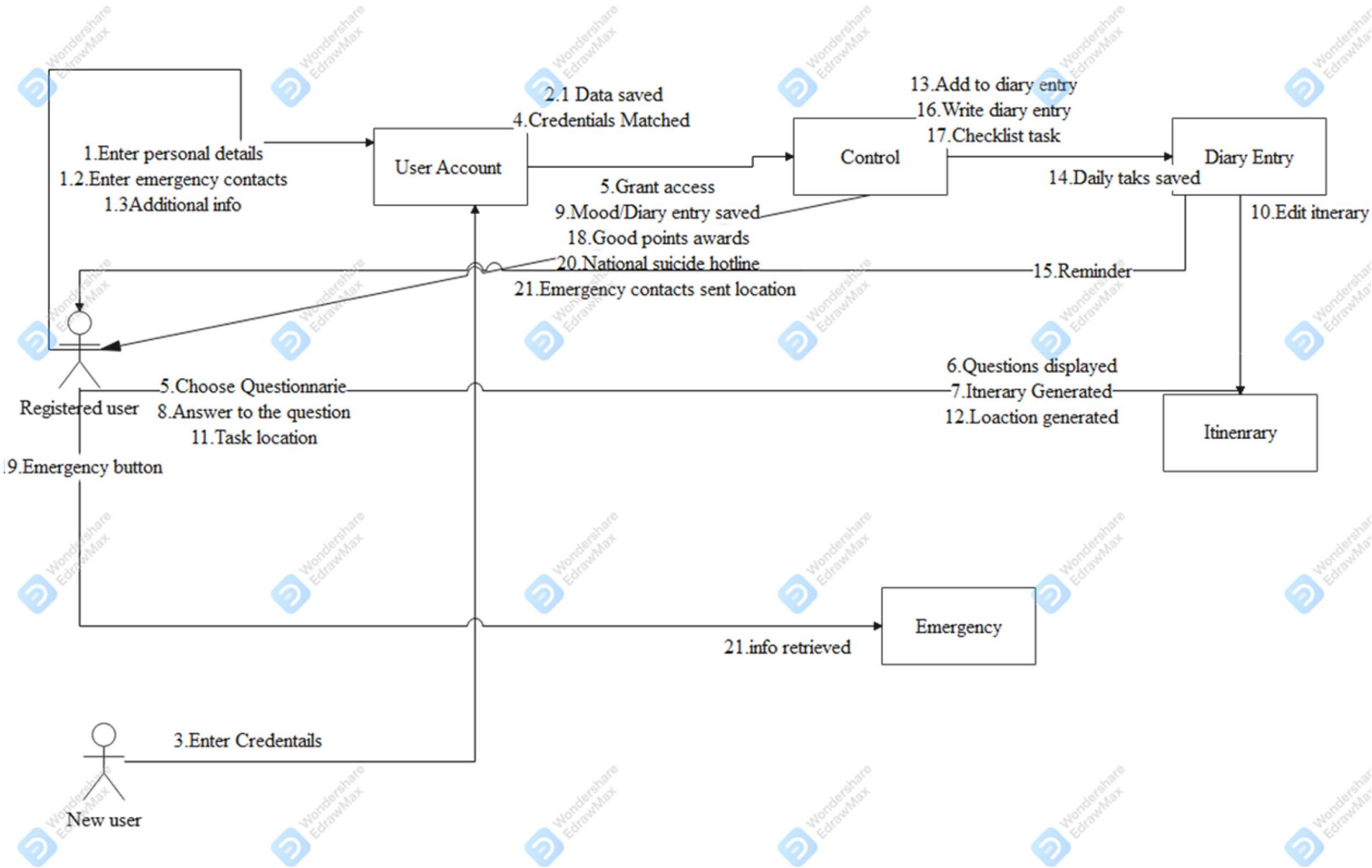
collaboration diagram

As the name suggests, the main focus of this type of UML diagram is on communication between objects.

Since the core components are the messages that are exchanged between objects, we can build communication diagrams the same way we would make a sequence diagram. Communication UML diagrams use number schemes and pointing arrows in order to depict the message flow.

Communication diagrams are much easier to design because you can literally add an object anywhere on the drawing board. After all, in order for objects to be connected, they only need to be part of the numbered sequence, without having to be physically close to each other.

Software used: EDraw max


1.Enter personal details
1.2.Enter emergency contacts
1.3Additional info
User Account
2.1 Data saved
4.Credentials Matched
Control
13.Add to diary entry
16.Write diary entry
17.Checklist task
Diary Entry
14.Daily taks saved
10.Edit itnerary
5.Grant access
9.Mood/Diary entry saved
18.Good points awards
20.National suicide hotline
21.Emergency contacts sent location
15.Reminder
Registered user
5.Choose Questionnarie
8.Answer to the question
11.Task location
6.Questions displayed
7.Itnerary Generated
12.Loaction generated
Itinenrary
9.Emergency button
Emergency
21.info retrieved
3.Enter Credentails
New user

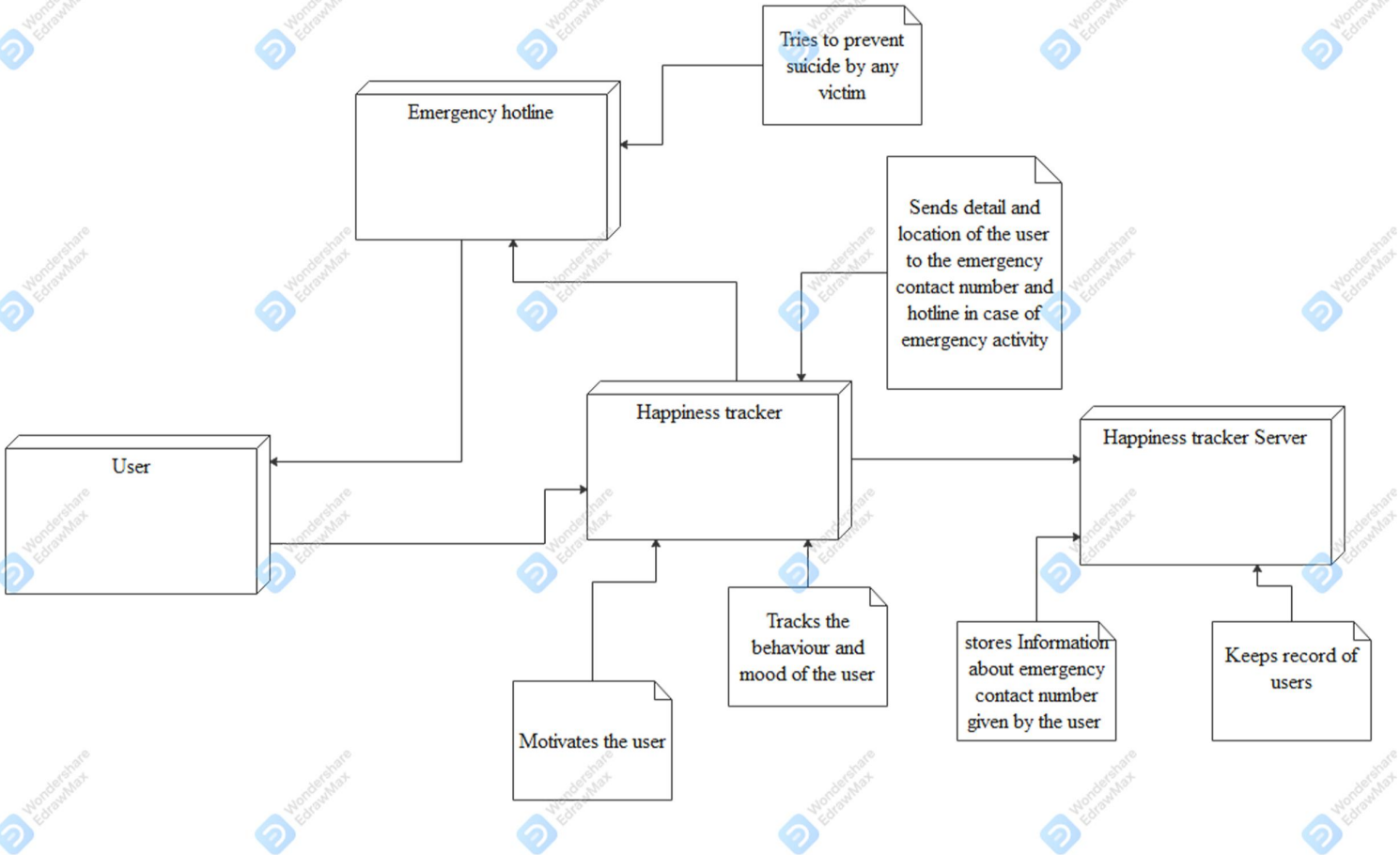
deployment diagram

Deployment diagrams are used to visualize the relation between software and hardware. To be more specific, with deployment diagrams we can construct a physical model of how software components (artifacts) are deployed on hardware components, known as nodes.

A typical simplified deployment diagram for a web application would include:

Nodes (application server and database server)
Artifacts (application client and database schema

Software used: EDraw max



component diagram

A component diagram displays the structural relationship of components of a software system. These are mostly used when working with complex systems with many components. Components communicate with each other using interfaces. The interfaces are linked using connectors.

Software used: EDraw max