Czech University of Life Sciences Prague

Faculty of Economics and Management

Department of Information Engineering

Diploma Thesis

Web-based Information System for the Tourist Visiting Nepal

Narvesh Pradhan

Declaration

I declare that I have worked on my diploma thesis titled "**Web-based Information System for the Tourist visiting Nepal**" by myself and I have used only the sources mentioned at the end of the thesis. As the author of the diploma thesis, I declare that the thesis does not break copyrights of any their person.

In Prague on 30/11/2020

Narvesh Pradhan

Acknowledgement

First of all, I would like to thank my family for their continuous encouragement, support and love. They have always been there for me in times of my need in all aspects of life.

I would like to show my gratitude to my thesis supervisor doc. Ing. Vojtěch Merunka, PhD of the Faculty of Economics and Management at Czech University of Life Sciences Prague for his great support and understanding in the completion of this thesis. Prof. Merunka was always clear and willing to answer whenever I had a question about my thesis research and writing.

I would also like to thank my teachers, my classmates and the academic staff of the Systems Engineering and Informatics of the Czech University of Life Sciences, Prague, who always supports me in the learning process.

Finally, I would like to thank my friends for their continuous support and motivation to complete the thesis. I also want to wish them a bright future and success in their life.

Web application for the Tourist visiting Nepal

Abstract

This project highlights the importance of the information system and its impact on the modern world. The development of web-based information over traditional information system has become valuable tools for any business organization to run effectively. It is the process for any small and big organization to organize, communicate and store data and information in a cost-effective and time-efficient manner. To design and model a robust information system, it is necessary to have a tool for initial planning, design, analysis and requirements. In this case, UML comes into rescue. In this web application, following the UML's footprint provides the idea about the initial planning, design, and implementation of a system with ease. It depicts the process of communication and the relationship between the application model. It is the standard for visualizing the architectural design of a system in a diagram. In the literature review, the brief history and origin of the UML along with software development life cycle is discussed. The software development phases gather the information, product description from the targeted customer following the preparation of hardware and software requirements. It defines the overall software architecture. The implementation phase is the developers translate into source code and finally testing and deployment are done after the software is free from errors and meets the requirements. Similarly, the other part describes the uses of UML diagrams that show the relationship between the elements and their communication. The two types of diagram, structural diagram show the structural or semantic relationship. On the other hand, the UML behavioural diagram depicts the dynamic behaviour of an element in a system that is dependent on time and how they relate to each other. After designing the prototype of an application, the next step is to build an actual application using WordPress. The primary focus of an application is to share information about popular tourist destination in Nepal. The prototype of the web application is developed using the class model, interaction model and state model and their respective UML diagram.

Keywords: UML, Unified Modelling Language, Object-oriented Design, Nepal Tourism, Web Application, Information Systems, Software Development Life Cycle, Content Management System, WordPress

Webová aplikace pro turisty navštěvující Nepál

Abstraktní

Tento projekt zdůrazňuje význam informačního systému a jeho dopad na moderní svět. Vývoj webových informací přes tradiční informační systém se stal cenným nástrojem pro efektivní fungování jakékoli obchodní organizace. Jedná se o proces pro každou malou i velkou organizaci, aby organizoval, komunikoval a ukládal data a informace nákladově efektivním a časově efektivním způsobem. Aby bylo možné navrhnout a modelovat robustní informační systém, je nutné mít k dispozici nástroj pro počáteční plánování, návrh, analýzu a požadavky. V tomto případě UML přijde na záchranu. V této webové aplikaci poskytuje sledování stopy UML jednoduchou představu o počátečním plánování, návrhu a implementaci systému. Zobrazuje proces komunikace a vztah mezi aplikačním modelem. Je to standard pro vizualizaci architektonického návrhu systému v diagramu. V přehledu literatury je diskutována stručná historie a původ UML spolu s životním cyklem vývoje softwaru. Fáze vývoje softwaru shromažďují informace a popis produktu od cíleného zákazníka po přípravě požadavků na hardware a software. Definuje celkovou softwarovou architekturu. Fáze implementace je vývojářem převedena do zdrojového kódu a nakonec je testování a nasazení provedeno poté, co software neobsahuje chyby a splňuje požadavky. Podobně druhá část popisuje použití UML diagramů, které ukazují vztah mezi prvky a jejich komunikací. Dva typy diagramu, strukturální diagram, ukazují strukturální nebo sémantický vztah. Na druhé straně diagram chování UML zobrazuje dynamické chování prvku v systému, které jsou závislé na čase a na tom, jak spolu souvisejí. Po návrhu prototypu aplikace je dalším krokem vytvoření a skutečná aplikace pomocí WordPressu. Primárním zaměřením aplikace je sdílení informací o oblíbeném turistickém cíli v Nepálu a poskytovani sluzebo a dene destinaci. Prototyp webové aplikace je vyvinut pomocí modelu třídy, modelu interakce a modelu stavu a jejich příslušného diagramu UML.

Klíčová slova: UML, Unifikovaný Modelovací Jazyk, objektově orientovaný design, Nepálská turistika, Webová aplikace, Informační systémy, Životní cyklus vývoje softwaru, Systém pro správu obsahu, WordPress

Contents

List of Figures:

1. Introduction

The development of technology has become a blessing for mankind. The breakthrough of the Internet and world wide web has brought us the new era of the world and has proven its impact on human life. We live in the digital world where the internet and information system are pivotal for the development. Moreover, information system enables people to communicate with each other despite the cultural gap and even linguistic barriers which have resulted in globalization and dispersion of ideas, new concepts.

The web-based information systems have evolved significantly in recent years. It uses internet web technologies for delivering information and services to users efficiently and cost-effectively. The web-based information system has gained its popularity since it allows the users to interact with a remote server through a web browser interface.

In this project, I have used WordPress and UML to design the web application for the tourist visiting Nepal. UML is used to develop the prototype for this application which come very handy to initial planning, analysis, design, and implementation in WordPress. I learnt UML is prerequisites for any software development. It is the standard for visualizing the architectural design of a system in a diagram. Furthermore, the UML reminds the software developers and architect to design a prototype for proper planning, design and whether it meets the requirements.

Web application for the tourist visiting Nepal is designed to give travellers the insights who are willing to visit Nepal. Travellers can just sit at home and explore the main tourist destination. The application can provide information about different places.

2. Objectives and Methodology

2.1 Objectives

The main objectives of this thesis are to design, implement, and test a web-based information system on miscellaneous tourist's services in Nepal. The system will have the UML implementation to show the relationship between different elements and their communication with each other. The purpose to use UML is to design the prototype for proper planning, analysis, design, and implementation of the system before writing the code itself. The system will have two access levels: the first level without the need to login and pay anything will be for providing basic information. The second level will be personalised to an individual user and will track their services and provide some feedback.

2.2 Methodology

To meet the requirements of the project a literature review on the information systems and UML will be carried out. The content management system i.e. WordPress is used to design the user interface and other functionalities. The login and register features, profile, booking services, feedback and other services are implemented using WordPress. Following the software development lifecycle, requirement analysis is performed from the users to meet future goals. Market research and reviews on other similar application are performed before implementing the project. First, the UML is used to design the prototype of an application that shows the communication between the different elements in the system and after the WordPress is used to design the user interface using the available plugins.

3. Literature Review

3.1 Information systems

The understanding of information system goes far beyond understanding technologies. Information systems are interrelated hardware and software components that organizations use to collect, store, process and disseminate information to support decision making, coordination, control, analysis, and visualization in an organization. Different kinds of components work together to provide value to an organization which are discussed below: (Bourgeois, 2014)

1. Hardware:

   The physical components like computers, keyboard, disk drives, and flash drives to which we can touch are the part of information systems hardware.

2. Software:

   Software is the intangible parts of information systems. Generally, the software is written set of instruction to guide hardware parts what to do and how to function. The software can be divided into system software i.e. operating system that manages the hardware, data and program files and application software, programs designed to handle specific tasks for users.

3. Data:

   Data are important components of information systems. These are the collection of facts. The organized collection of data permits automated sentiment analysis for marketing, competitive intelligence, new product development and another decision-making process.

4. People:

   The contribution of people includes development and operation managers, business analysts, system analyst and designers, database administrators, programmers must not be overlooked to make an information system. after all, people are the one who is capable to integrate into an organization with the other components of information system. (Bourgeois, 2014)

5. Process:

   Processing the data and information is taken as the valuable component achieve a desired outcome or goal. Information systems are bringing more productivity and better control to the organizational process.

3.2 Unified Modelling Language

The architecture and development of Unified Modelling Language have brought a huge impact in the field of software development. The complication in the system in this technological era can be tedious to manage or develop the featured system but following the UML's footprint provides the idea about the initial planning, design, and implementation of the system with ease. It is the standard for visualizing the architectural design of a system in a diagram. Furthermore, the UML reminds the software developers and architect to design a prototype for proper planning, design and whether it meets the requirements. The UML provides software developers, system architect and software engineers with necessary tools for analysis, design and implementation of the software-based system as well as modelling the business and similar process. The system without planning and that cannot meet the future requirement can be fragile. In this case, UML comes handy because it offers system blueprints to plan, design, analyse, and implement effectively. (Object Management Group, 2017)

UML is a general-purpose modelling language which helps to simplify the process of software design. The use of graphical notation to create visual models of an object-oriented software system. UML is the standardized modelling language that is intended to provide the standard way to specify, visualize, construct and document artefacts of a software system. Simply, it makes these artefacts robust, scalable and secure in execution. One of the main goals of UML is to provide software developers with tools for analysis, design and implementation of a software system as well as another similar process. It uses a graphical notation to create a visual model of software systems. (Technopedia, 2019)

UML may be a standardized modelling language for software developers to explain, specify, design and document existing or new business processes, or behaviour of artefacts of a software. UML specification 1.4.2 has explained the following process:

- Helps to specify what artefacts should be developed
- Guide as to the order of the team's activities
- Helps to manage and directs the task of individual developers and the team as a whole

  (Fakhroutdinov, 2019)

3.2.1 History of UML

As systems become more complex, the development of a software system is more difficult in planning, designing, implementation and deployment. On the other hand, system maintenance and re-usability also become more and more important. In 1980s the first object- oriented modelling language was emerged along with the development of object-oriented language. Until about 1970s software development is just an artistic venture. So, when the system becomes more complex, the development and maintenance could no longer be covered with this creative-individual approach. Thus, lead to the software crisis. In 1990s object-oriented analysis and object-oriented design has been published for the first time and further 50 object-oriented methods were implemented as well as many design formats. The unified modelling language seemed crucial for software development. (sourcemaking, 2019)

The main motivation behind developing the UML is to standardise the disparate notational systems and approaches to software design. In 1994-95, Grady Booch, Ivar Jacobson, and James Rumbaugh coordinated together to develop UML and was adopted by OMG (Object Management Group) as a Standard in 1997. OMG's main goal with method and data that work using all types of development environments on all types of platforms was a common portable and interoperable object. The creation of unified modelling language was created upon standardization of notation that seemed less elaborate than the standardization of methods, They integrated the Booch method of Gary Booch, Object Modelling Technique of James Rumbaugh and Object-Oriented Software Engineering (OOSE) by Ivar Jacobson and published a new notation name UML version 0.9. The main objective is not to design or formulate the new notation but to adapt, expand and simplify the existed and accepted types of diagrams of several object-oriented methods such as class diagram, Jacobson's use case diagram, or Harrell's State chart Diagram. (source making, 2019)

Gary Brooch's technique is a predecessor to UML. It is a widely used technique in software development for object-oriented analysis and design. This helps to covers the analysis and design phase of object-oriented system implementation. The elements of Object-Modelling

Technique (OMT) and Object-oriented Software Engineering (OOSE) and graphical elements of Booch method were featured as the notation aspect of Booch method.

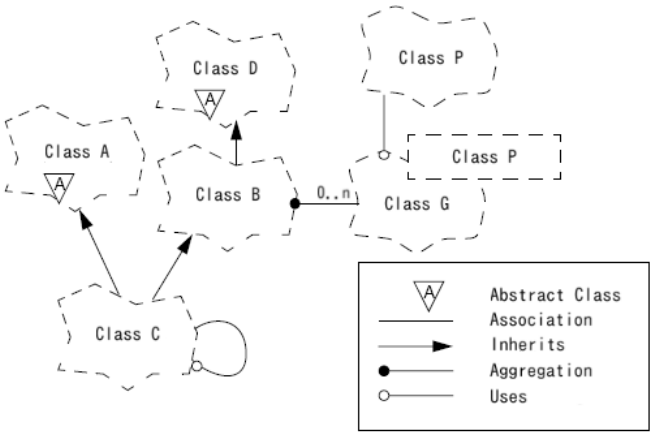

Figure 1 Gary Booch OO analysis and design

Ivar Jacobson's OOSE model includes a requirement, an analysis, a design, and implementation and a testing model is the first object-oriented design methodology to employ use cases to drive the software design and development. The use cases describe the complete functionality of the system and serve as the central model of the system. Its emphasis on analysis, construction and testing phase of any software system. The key to the analysis phase is to identify the objects and their interactions. The construction phase is mainly concerned with the design and implementation in source code and components are used to implement objects. (Jacobson, 2013)

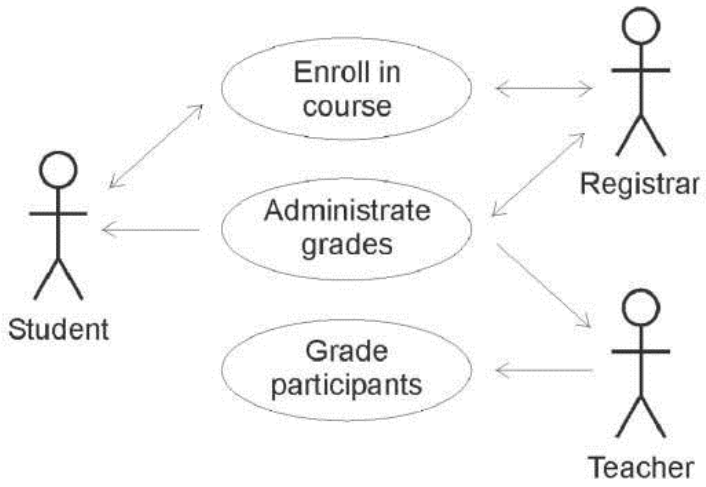

Figure 2: Use case diagram for student enrol application

James Rumbaugh's Object Modelling Technique is an approach to object-oriented development in analysis and design phase. The analysis phase begins with a problem statement which includes a list of goals of a system that can fulfil the future requirements. Object, dynamic and functional model are further divided into three models as a problem statement. System Artifacts are described in the object model whereas the dynamic model represents the interaction between the artefacts. The analysis phase is complete with a functional model that represents methods of a system from a perspective of data flow. The design phase established the overall architecture of a system. Here, the whole system is divided into sub-system to allocate process and task keeping in mind about the system's concurrency and collaborations. (Burback, 1998)

In 1996, a consortium called UML Partners was organized under the technical leadership of Booch, Jacobson and Rumbaugh to complete Unified Modelling Language specification and propose it to the Object Management Group for standardization. The first release of the UML 1.1 was introduced by OMG in 1997 with combined efforts of IBM, Microsoft and Oracle after the final specification and integrate it with other standardization efforts. New version UML 2.0 was introduced in 2005 which provides many features like external user interface, component and interface interaction and system activities. Moreover, the UML version 2.x specification has the supermodel that defines the notation and semantics for diagrams and their model elements. The structure that defines the core metamodel on which the superstructure is based and the UML diagram that defines how the UML2 diagram layout is exchanged. (Wikipedia, 2019)`

3.3 Software Development Life Cycle

Software development goes through the series of process from requirement analysis to deployment and maintenance to meet the required objectives. It is used to construct software determining the capabilities it has, how it is constructed, who works on what and the time frames of all activities. There are following six phases in the software development life cycle model:

- Requirement gathering and analysis
- Design
- Implementation
- Testing
- Deployment
- Maintenance

3.3.1 Requirement gathering and analysis:

In this phase, the essential information is collected from the customer to develop as a product as per their expectation. Before building the product the core understanding of the product is important. So, it is necessary to organize a meeting with the customer to gather information like what the customer wants to build, who will be the end-user and what is the purpose of the product. (softwareTestingHelp, 2017)

3.3.2 Design:

The system and software design are prepared after the requirement analysis specification which is studied in the first phase. In this phase the hardware and software requirements are specified, and overall software architecture is defined.

3.3.3 Implementation:

This phase is the longest in the software development life cycle. The tasks are divided into smaller modules after receiving the software design document and are translated into source code by the developers. (softwareTestingHelp, 2017)

3.3.4 Testing:

Testing starts once the coding is complete and is tested against the requirement to make sure that the product is solving the needs addressed and gathered during the requirements phase. This process ensures that the software is free from errors.

3.3.5 Deployment:

After successful testing of the product is delivered, it is deployed in the production environment or first UAT (User Acceptance Testing) is done depending on the customer expectation.

3.3.6 Maintenance:

The main objective of software maintenance is to make the software operational as per the user requirements and to provide continuity of the service.

The process of software development in UML follows the process like inception, elaboration, construction, and transition incrementally and iteratively. Rational Unified Process (RUP) is a software development process framework developed by Rational Corporation describes the properties like use case is driven, incremental and iterative, and architecture-centric to get most out of UML. The construction phase consists of many iterations, each iteration builds production-quality software, tested and integrated, that satisfies the subset of the requirements of the project. Each iteration is similar to the software development life cycle phase of analysis, design, implementation, and testing. During inception, the scopes of the project are described, and the business rationale of the project is explained. The project goals, features, functions, deliverables and ultimately cost can be included here. In elaboration, plans for the construction for the project are executed by collecting more detailed requirements and doing a high level of analysis and design. Even most of the task is done in each iteration but still, beta testing, performance tuning, and user training are performed in a transition phase. (Fowler, UML Distilled: a brief guide to standard object modelling language, 2000)

3.4 UML Diagrams

UML diagram represents the partial graphical view of a model of a system under design, implementation or already in existence. UML contains the graphical elements(symbols). The use of different diagram from the initial planning phase to modelling, maintenance of the whole system can meet the future requirement of a project. Developers can use a use case diagram on how the system function before designing the actual system itself to class diagram that describes the structure of the system and so on. The use of graphical notation in UML is to create visual models for design, analyse and implement different system process. However, it has some drawbacks that it does not have feature mixing different types of diagram e.g. to combine structure and behavioural elements to show a state machine nested inside a use case. UML specification has two major types of diagrams:

- Structural diagrams, and
- Behaviour diagrams

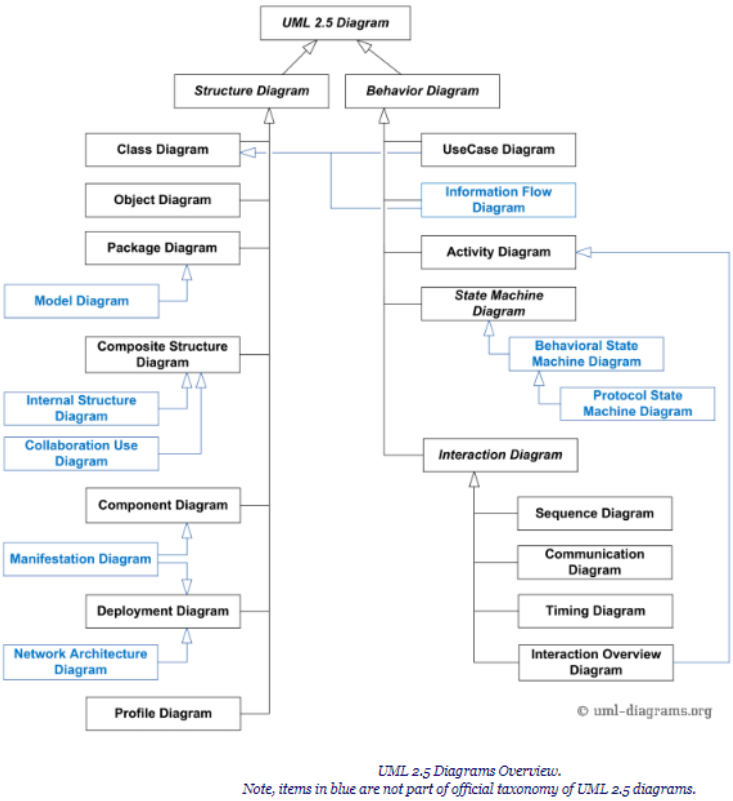

Figure 3 types of UML Diagrams

Modelling Architectural Views using UML

In a real-world system, a user can be developers, testers, businesspeople, analyst and many more. It is always important to visualize the system from different viewer's perspective that leads to successful design, implement and deploy a system to the real world. This set of views is called the 4 + 1 views of software architecture. And, UML has an important role in defining different perspective of a system which are as follows:

- Use case view
- Design
- Implementation
- Process and
- Deployment

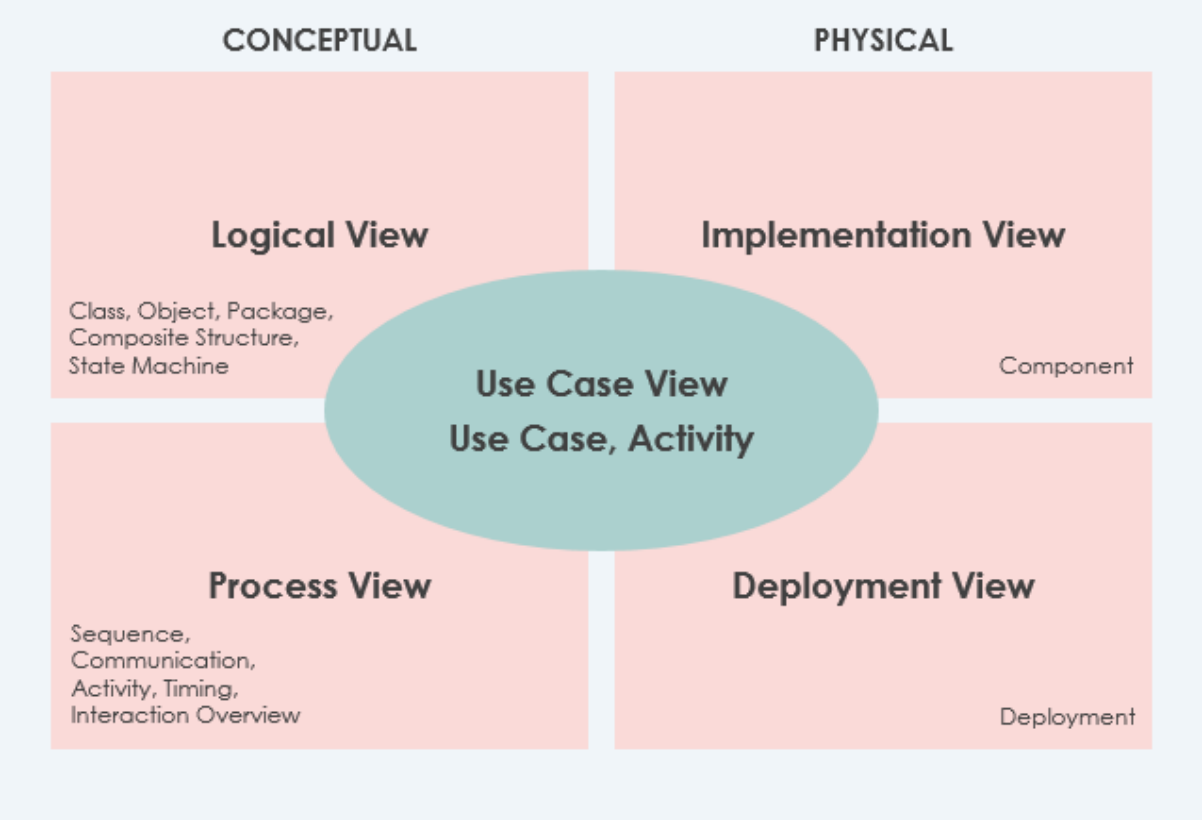

Figure 4 Architectural Views of UML diagrams (VisualParadigm, 2017)

The use case plays a central role by connecting all these four views.

- Use-case view: it illustrates the functionality of a system, it's principal users (actors) and the relationship between use case and actors.
- Logical view: UML diagram like class and state diagrams represents the logical view of a system. it describes the functionality that a system provides to the end-users.
- Process view: it focusses on the run time behaviour of the system, explain the process and how they communicate. It is the dynamic aspect of the system. Example: activity diagram.
- Development view: the development view also known as implementation view concerned with software management. It illustrates the system from the programme's perspective.
- Physical view: system engineers are more concerned about the physical view which depicts the topology of software components on the physical layer as well as the physical connection between the components. Deployment diagram is used to illustrate the physical view of a system.

3.4.1 Structural Diagrams

Structural diagrams feature the static structure of a system that convey the system concept in time independent manner and their relationship with each other. The elements in the structural diagram always show the structural or semantic relationship. These diagrams do not show the details of the dynamic behaviour and are not utilizing the time-related concept. (Fakhroutdinov, 2019) Structural diagrams include diagrams like class, object, package, composite structure, component and deployment diagrams. These diagrams visualize how the system function from the initial input to processing and finally to the desired output.

Class Diagram

No doubt the class diagram is widely used and important subject to the greatest range of modelling concepts. It captures all the logical structures of the system i.e. classes and thing that make up a model. Its static structure describes what exists and what attributes and behaviour it has, rather than how something is done.

To visualize the relationship between the components, their structure and communication through the paths such as association, aggregation, and generalization which are valuable for reflecting the inheritance, composition or usage and connections, a class diagram is always useful. It depicts the static structure of a system at the level of classes and interfaces, shows their features, constraint and relationship. It has become a useful tool for software developers for modelling a simple or complex business process. It helps to visually express any specific needs of a system and disseminate that information through the business. (SparxSystem, 2000)

So far, we know the class diagram summarizes the types of object in the system and the different kinds of static relationship that exist among them. It holds the properties and operation of a class and the constraints that apply to the way objects are connected. Moreover, to show the relationship between the classes and interfaces paths like a generalization, association and aggregation are used which are valuable for reflecting inheritance, uses and connections respectively.

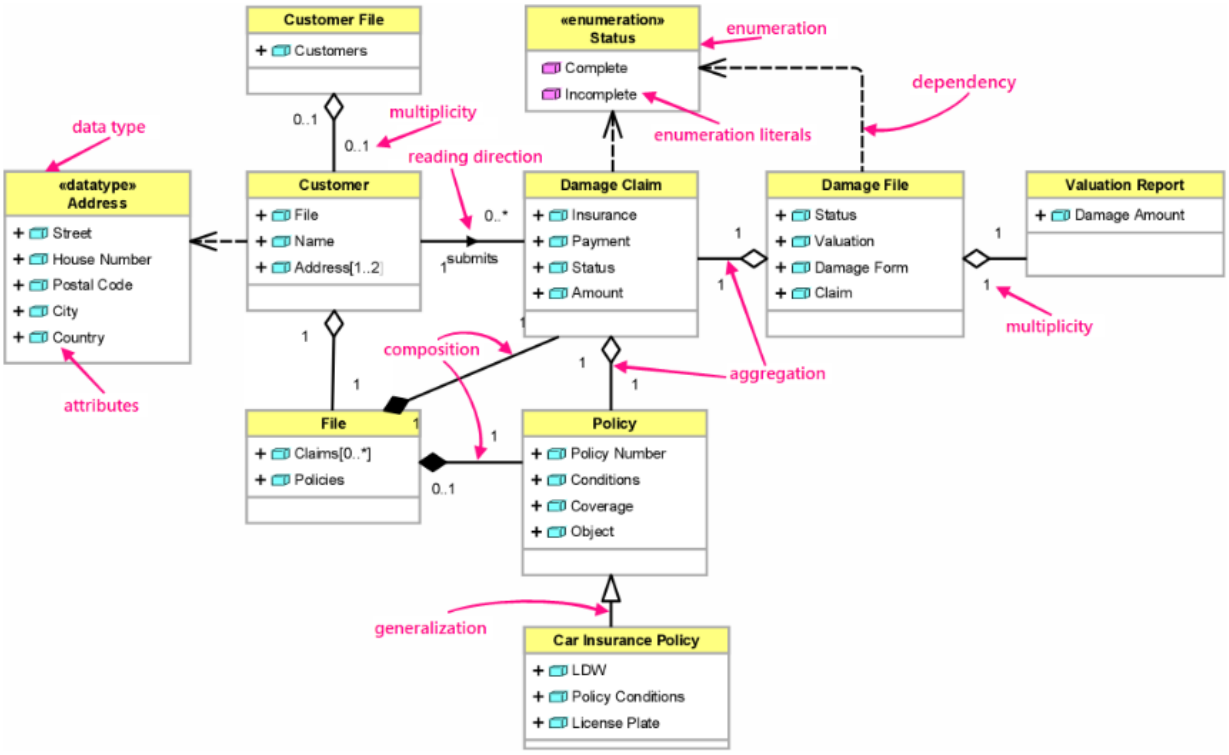

Figure 5 class diagram and their relationships (Bizzdesign, 2017)

The above figure shows the static structure of a system by imaging the classes of a system, their attributes and operation and relationship between the classes.

Association

Association is a way to notate a property in a class. If two classes in a model need to communicate with each other, a link is created which is represented as an association. For example, A student and faculty are having an association, meaning one can be in existence if other is demolished.

Association class

An association class is an association relationship between two classes that provides additional information about the relationship. Association class is identical to other classes and can contain operation, attributes as well as other associations. (IBM, 2015)

Aggregation and Composition

These are the special form of association relationship between the classes. Aggregation is a part-of relationship. An aggregation relationship can be defined as 'an object of one class can own or access by the object of another class'. It establishes strong ownership between the object meaning that the dependent object remains in the scope of relationship even when the source object is destroyed. Composition, on the other hand, is a good way of showing properties that own by value, properties to value objects. It is just like an aggregation except the 'whole object' controlled the 'art object'. So, if a composition is destroyed, all its parts are destroyed with it, however, a part can be destroyed from the composition without having to destroy the entire composition. The most important concept to remember in the composition is that component instance is part of at most aggregate instance, i.e. it cannot be shared between multiple instances. (Flower, 2004)

Classification and Generalization

The author of book UML Distilled 2.0, Martin Flower provides a suitable example to understand classification and generalization. He urges to beware about the subtyping as the is a relationship. This can be meant different things. Let us consider the following phrases.

1. Sheep is Border Collie
2. A Border Collie is a dog.
3. Dogs are animals.
4. A Border Collie is a Breed

5. The dog is a Species.

Trying to combine the phrases 1 and 2, we get 'Shep is a Dog.' 2 and 3 yields 'Border Collies are Animal.' These make sense but combining 1 and 4: 'Shep is a Breed.' 2 and 5 is 'A Border Collie is a species.' These are not so good. We found out that combining these phrases some make sense but not others. The reason is some are classification, and some are a generalization. From the above phrases, the object Shep is an instance of the type Border Collie and the type Border Collie is a subtype of the type Dog is classification and generalization respectively. (Fowler, UML Distilled 3rd edition, 2003)

Generalization is a simple yet very important concept that shows the parent and child class relationships. Generally, the child inherits the attributes, operation and relationships that are defined in the parent and reuse them in one or more child model elements. This way, it is not necessary to write all the attributes and operation shared between them. In the generalization relationship, the parent model can have one or more children and vice versa.

Object Diagram

An object diagram is similar to the class diagram which represents the specific instance of the class diagram in a particular moment. It primarily focuses on the attributes of a set of objects and how those objects align with each other.

It can be used as the special case of a class diagram, they do not architecturally different than class diagram but reveal the multiplicity and function of a diagram. Object diagram makes use of a subset of an element of a class diagram to be able to highlight the relationships between the instance of classes in the future. (EDUCBA, 2019)

Composite Strucutre Diagram

These diagrams are the new concept of UML structural diagrams that shows the internal structure including parts and connectors of a structured classifier or collaboration. A composite diagram describes the detail internal structure of multiple classes and the interaction between them. It contains classes, packages, and interfaces, their relationships and provides the logical view of all or part of a system.

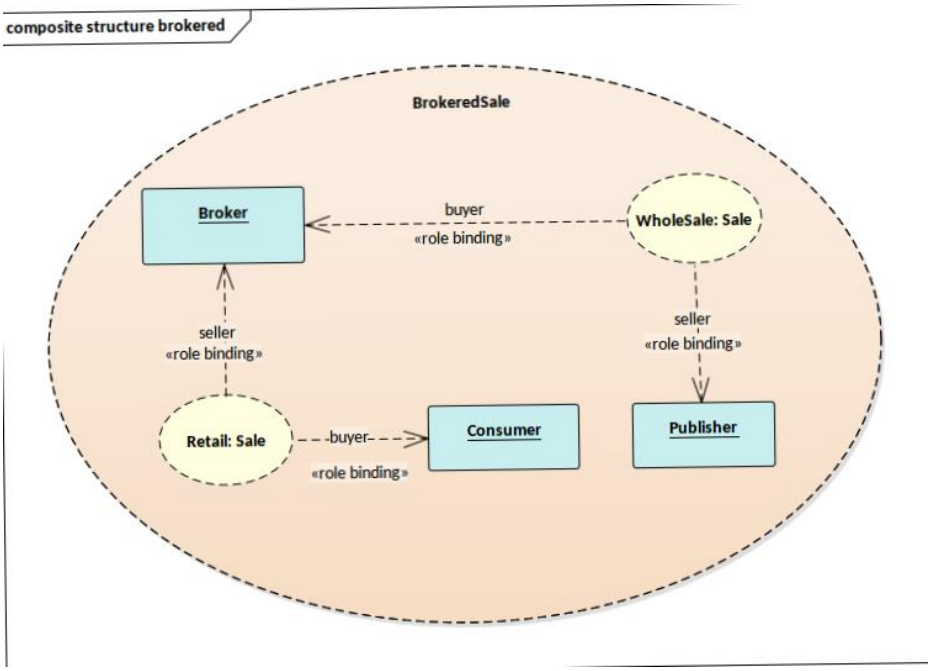

Figure 6 composite structure diagram (SparxSystem, 2000)

Deployment Diagram

The term deployment describes the hardware components where the software components are deployed. The deployment diagrams are the types of UML diagrams that represent the run time architecture of a particular systém. It consists of a set of nodes which are interconnected to each other and often helps to visualize the overall deployment of a systém. Moreover, these diagrams show the configuration of hardware components (nodes) of a particular systém, their interconnection and shows the software components and artefacts are mapped onto those nodes. Deployment diagram model the hardware topology of systém rather than the logical component of systém. Thus, the deployment diagram not only helps us to visualize the hardware and software components of a system but also helps to picture the system as a whole. Systém engineers are greatly benefited because it shows how the hardware components are interrelated to each other and how the software component resides with the hardware component of a systém. (EDUCBA, 2019)

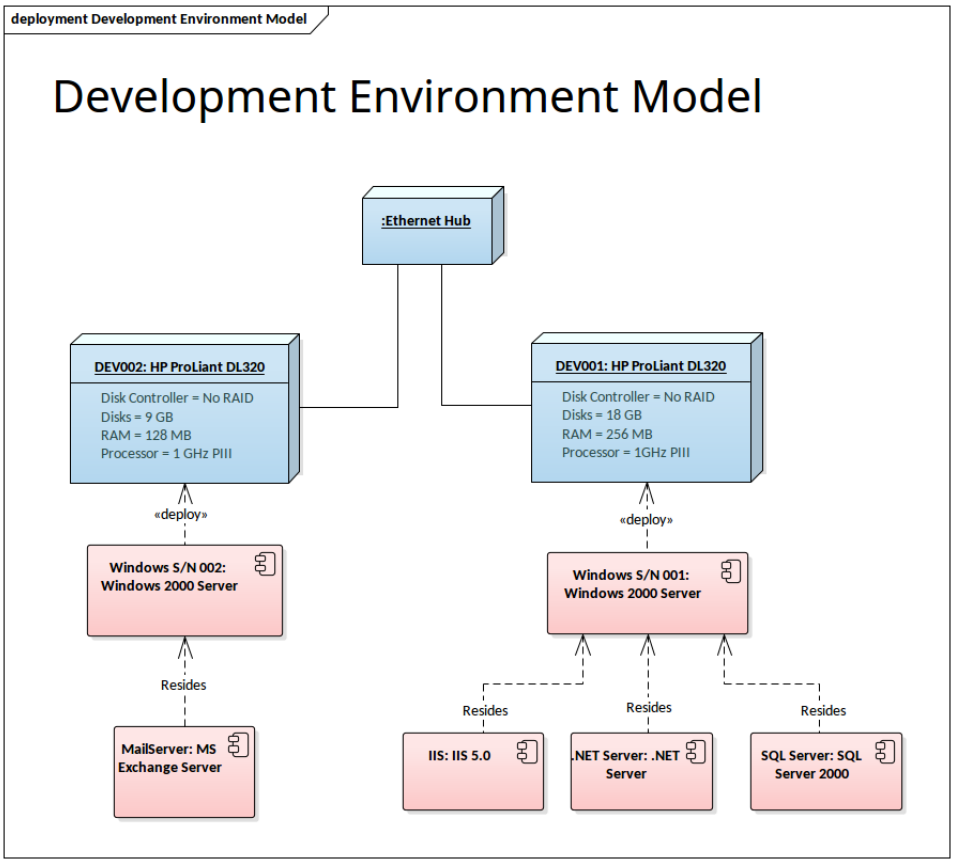

Figure 7 Deployment Diagram(SparxSystem, 2000)

Package Diagram

These types of static structure diagrams are useful to show the packages of classes that make up the model and the dependencies between the packages. It helps to simplify the complex class diagram structure. It can use packages that represent the different layers of a software system to illustrate the layered architecture of a software system.

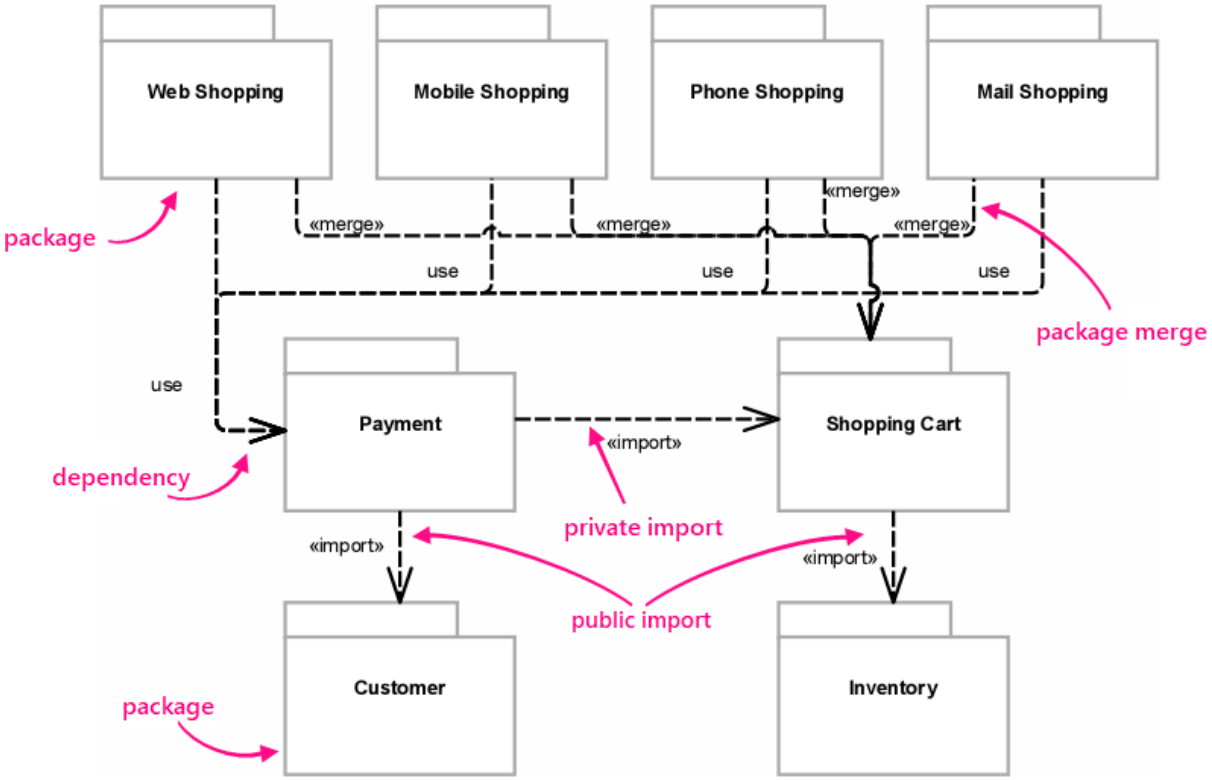

Figure 8 Elements of package diagram (Bizzdesign, 2017)

3.4.2 Behavioural Diagrams

UML behavioural diagram depicts the dynamic behaviour of an element in a system that is dependent on time and how they relate to each other. As it executes over time, it captures the various kinds of interaction and instantaneous sate within the model. Interaction model relates to the part of an application on how an object and action are interrelated in a way to support user interaction. It encompasses the exchange of messages between the set of objects in a specific situation to achieve a definite purpose. Secondly, the state model defines the sequence of operation that occurs in response to external stimuli. (SparxSystem, 2007)

Use case diagrams, activity diagram, sequence diagram, State machine diagram, communication diagram, interaction overview diagram, timing diagram are some of the types of behavioural diagrams. They all feature the dynamic behaviour of a system i.e. inner flow of messages, their relationship and functions in a certain period.

Use Case Diagram

Use case diagrams are especial diagram for organizing and modelling the behaviour of the system. It shows the functional requirements of a system describing the interaction between the users and the system. A use case can be described as the communication between the actors and other stakeholders about what a system is intended to do. A use case can also be described as a set of scenarios bound together by a common user goal. A scenario is a sequence of steps showing the interaction between a user and a system. In most cases, the user is referred to as an actor. It is a role played by the user concerning the system. A single actor may perform several use cases, or a use case may have several actors performing it. (Fowler, UML Distilled 3rd ediion, 2003)

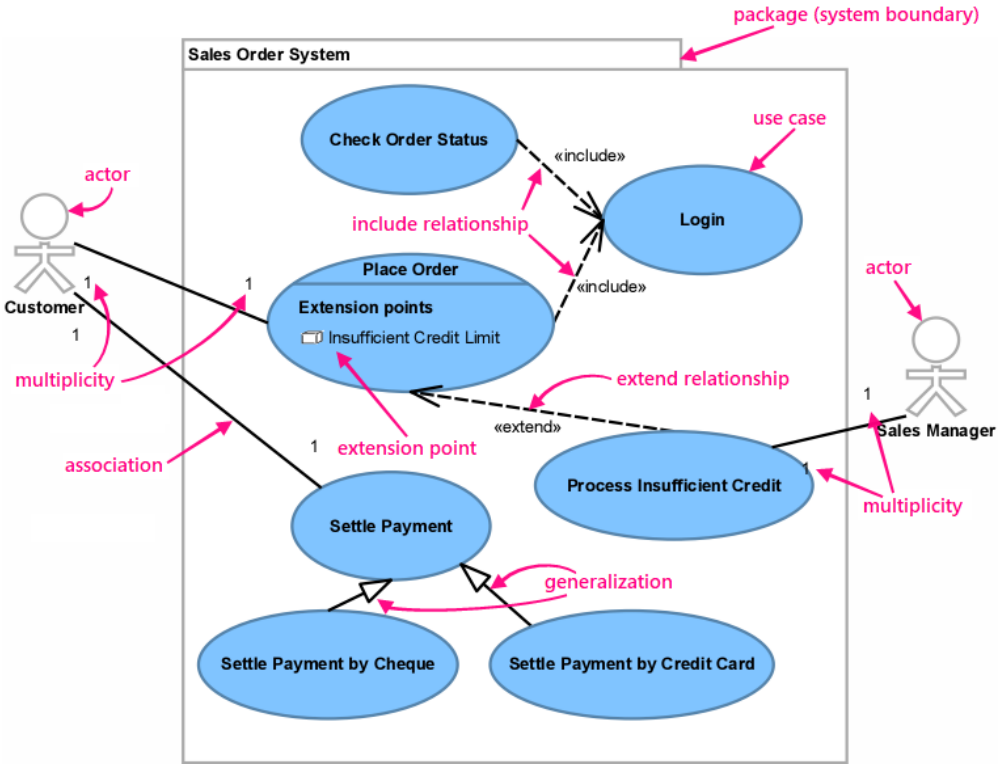

Figure 9 use case diagram and its interaction

The above figure is an example of a use case that performs the dynamic behaviour of a system that describes the set of actions that a system should perform in a collaboration with the external users of a system. It shows different types of relationships to communicate with different scenarios: Include, Extend and Generalization. An include relationship in the use case is the ability to include or to use other use cases in a clearly defined place in its explanation. The extend relationship in use case means the base use case is extended with additional behaviour by the extending use case. We use use case generalization when we have one use case that is like another use case but does a bit more. It provides an opportunity to capture alternative scenarios.

However, just focusing on the interaction between the user and the system can neglect the situations. This is the most common problem with the use case and change to the business process may be the best way to deal with the problem. (Fowler, 2003)

Sequence Diagram

These types of diagrams are the most common types of diagrams among interaction diagram which describes how groups of objects collaborate in some behaviour. Sequence diagrams are not designed to regulate complex procedural logic in a system, it shows which object communicate with which other objects, and what message trigger those communications. A sequence diagram consists of different elements like lifelines which represents an individual participant containing its object name in a rectangle. In the same way, arrows are used to display the messages. It can be lost and found; synchronous or asynchronous; call or signal. (Fowler, UML Distilled, 3rd Edition, 2003)

The following example of a sequence diagram depicts the interaction between participants with a lifeline that runs vertically down the page and the ordering of messages by reading down the page. The author Martin Fowler suggests using the sequence diagram when we want to look at the behaviour of several objects within a single use case. They are good at showing the interaction logic between the objects in a system in the time order that the interaction takes place.

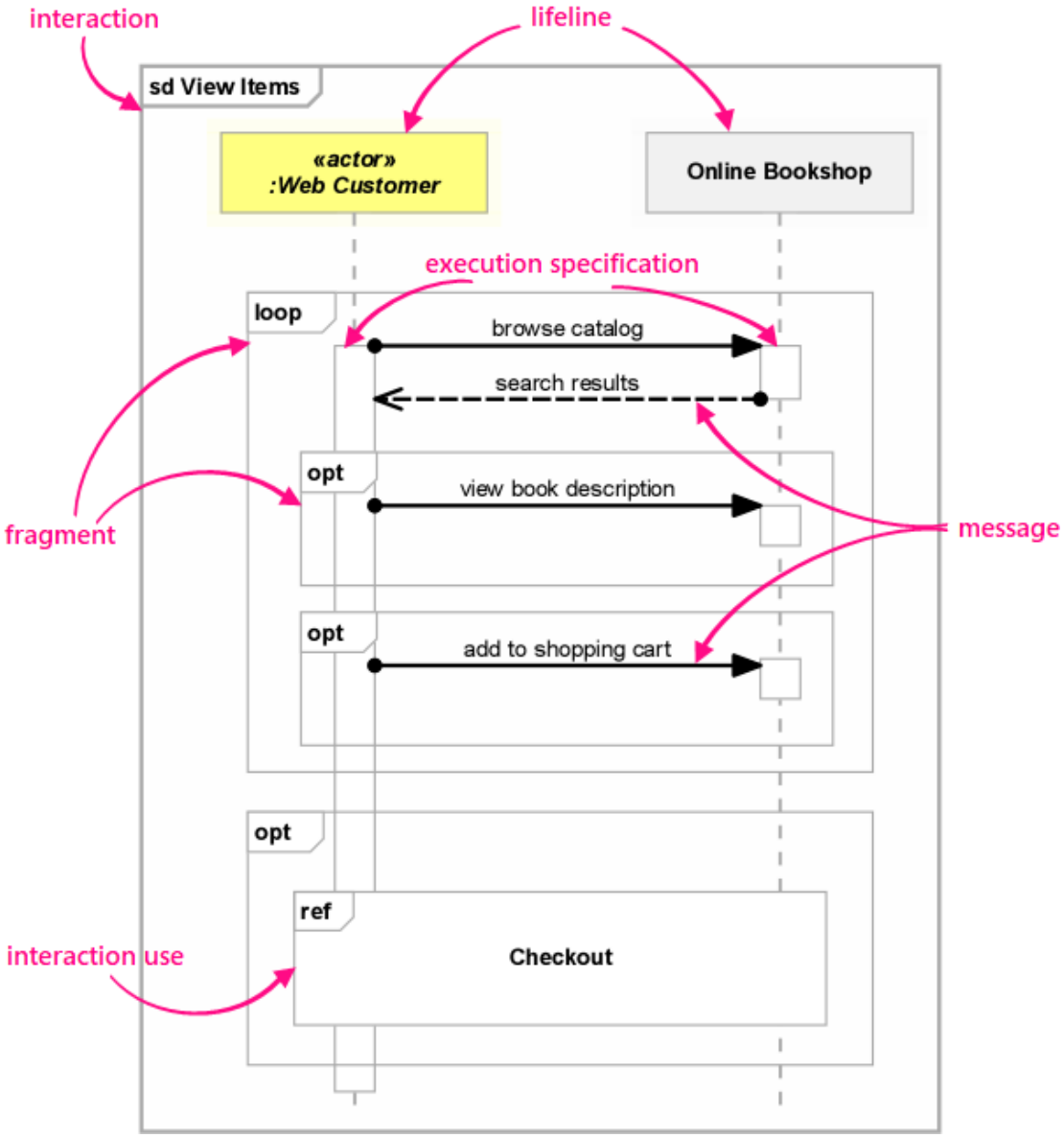

Figure 10 Elements of sequence diagram (Bizzdesign, 2017)

Activity Diagram

Activity diagrams are the dynamic modelling solution in UML that display the sequence of activities of a system. these behaviour diagrams show the flow of control or object flow with emphasis on sequence and condition of the flow. The workflow of the details of decisions path that exists in the progression of events from the start point to end point can

be shown in an activity. It consists of different elements from the start point to an endpoint behaving differently. (SparxSystem, 2007)

It can be regarded as the techniques in UML to describe the business process, procedural logic, and workflow of a system. The graphical representation plays a similar role as a flowchart. The only difference between the flowchart with an activity diagram is that the activity diagram support parallel behaviour. Activities, actions, control flow, a start node, end node are some of the elements present in an activity diagram. In UML, activities represent the specification of the parameterized sequence of behaviour. Thus the great strength of using activity diagram is that it support and encourage the parallel behaviour that leads to a great tool for workflow and process modelling. (Fowler, UML Distilled, 3rd Edition, 2003)

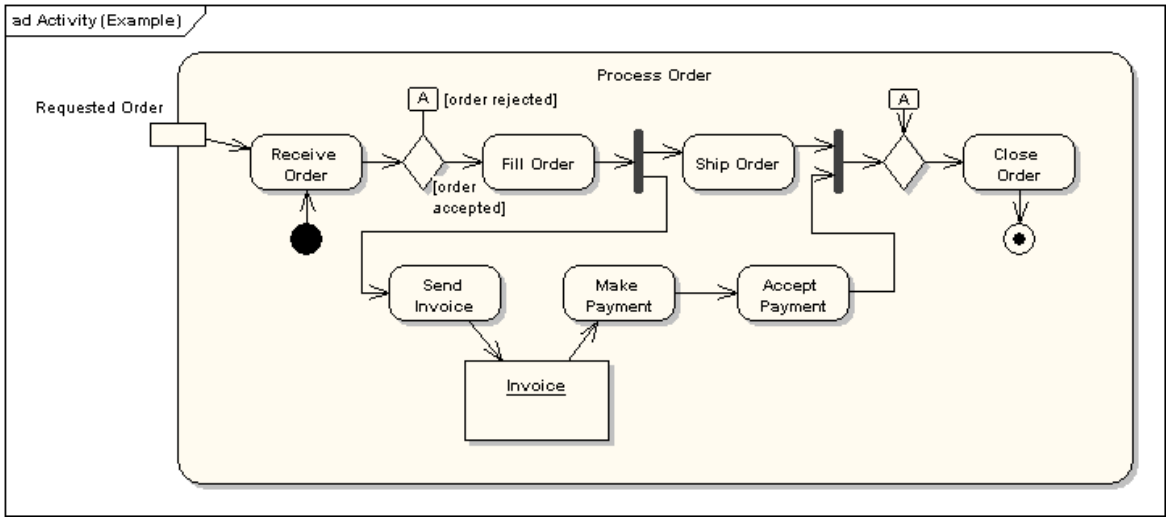

Figure 11 elements that constitute the activity diagram

State Machine Diagram

State machine diagram is the techniques in UML that describe the behaviour of an object across several use cases. However, it has got a limitation on not describing behaviour that involves several objects collaborating. So there are other techniques such as interaction diagram which is a good way to describe the behaviour of several objects in a single-use case, and an activity diagram which describe the general sequence of activities for multiple objects and use cases. (Fowler, UML Distilled, 3rd Edition, 2003)

These diagrams describe the dynamic behaviour of a system which shows the discrete behaviour of the designed system through finite state transitions. These techniques are often used in software development to show the behaviour of a system at different levels such as a class, a subsystem or an entire application. One of the most important things to remember is the state machine diagram is that it shows only what object directly observes or activates. The state is drawn as a round-cornered rectangle with the name inside it. It goes through the transition triggering different state actions. A transition from one state to another is denoted by lines with an arrowhead. (SparxSystem, 2007)

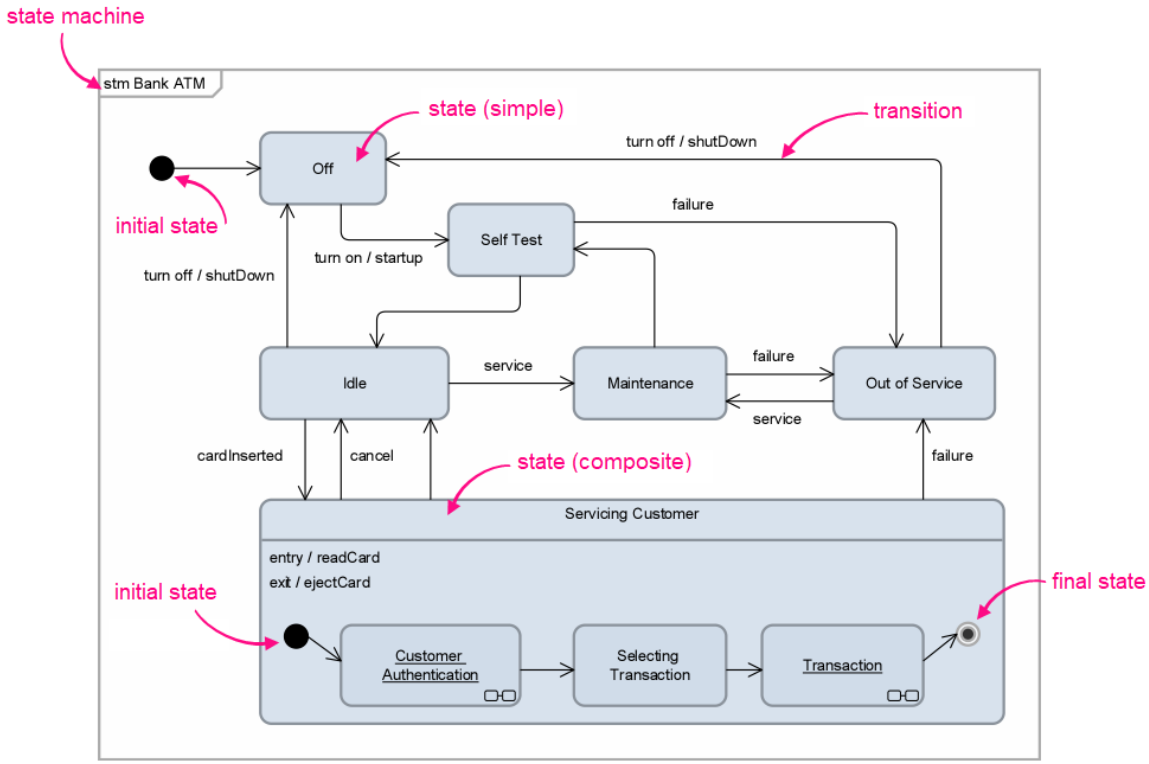

Figure 12 elements of a state diagram (Bizzdesign, 2017)

Interaction Overview Diagram

Interaction overview diagram is a similar technique as an activity diagram in which the activities are replaced by little sequence diagram, or as a sequence diagram broken up with an activity diagram notation used to show control flow. This diagram is a combination of activity and sequence diagram. This is the new technique in UML 2, it mainly focuses on the control flow of interactions which can also show the flow of activity

between diagrams. In contrast to an activity diagram, each activity is regarded as a frame which can contain a nested interaction diagram. This makes the interaction overview diagram useful to deconstruct a complex scenario to be illustrated as a single sequence diagram otherwise that would require multiple if-then-else paths.

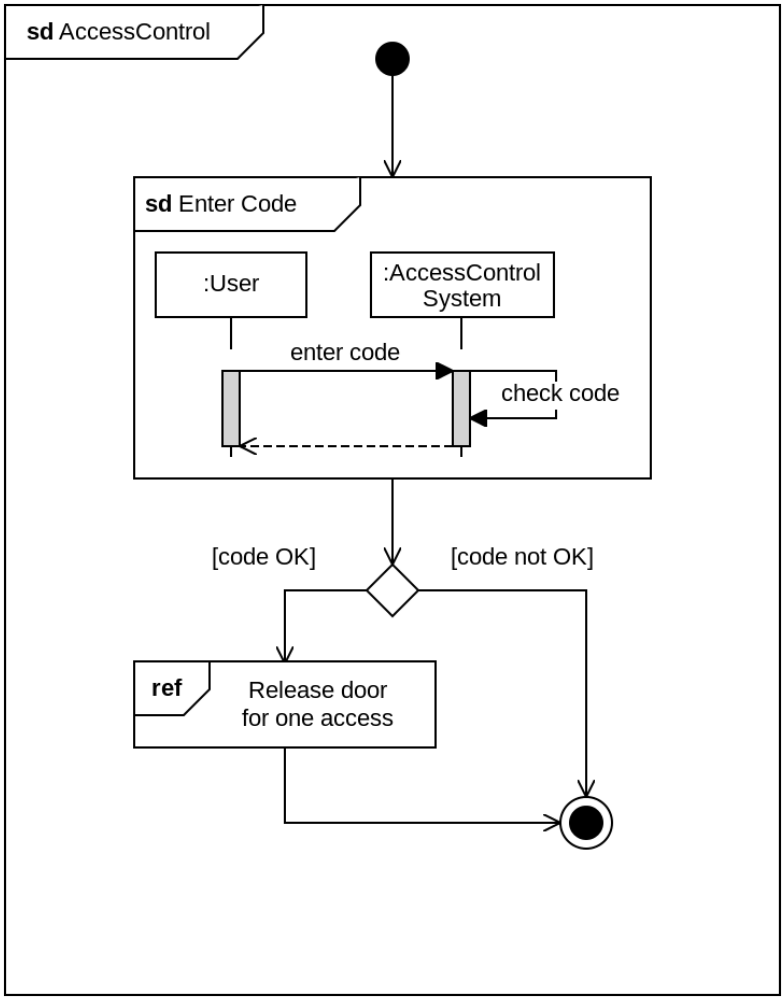

Figure 13 Interaction overview diagram *(wikipedia, 2019)*

Communication Diagram

The communication diagram is a type of interaction diagram that models the interactions between various participants in terms of sequence messages. It can describe both static and dynamic structure of a system as it represents the combination of class, sequence and use case diagram. These diagrams are helpful because it allows free placement of

participants, draw a link to show how the participants connect and use numbering to show the sequence messages. In contrast to sequence diagram, it does not need to draw a lifeline for each participant and show the vertical direction for a sequence of message flow, but it can offer the software engineers for the free placement of participants. (Fowler, UML Distilled 3rd edition, 2003)

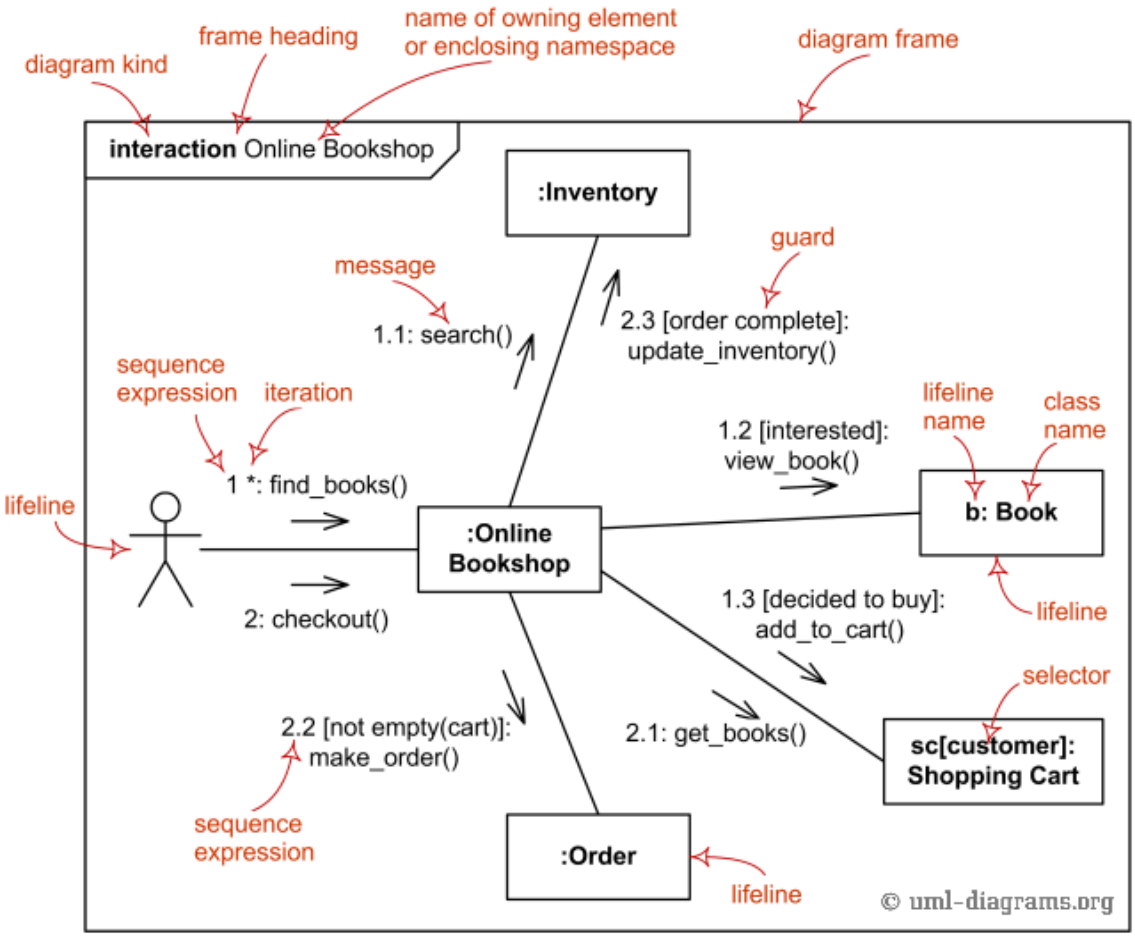

Figure 14 major elements of the communication diagram ***(Fakhroutdinov, 2019)***

The above figure shows the communication diagram for online bookshop as centralized control. All the participants in this diagram are linked together that are instance of an association. Although the communication diagram doesn't offer any precise notation for control logic. They have the feature to use iteration markers and guards. These are already dropped from UML 1.0 but still can be used in the communication diagram.

4. Object-Oriented Design in UML

In the other part of my project, I would like to focus on the object-oriented methods in UML. It is safe to say UML is designed and modelled around object-oriented modelling language. The UML history is a proof its initial versions were based on object-oriented methods – Booch, OMT and OOSE. Object-oriented can be a process to design and implement a system as a collection of interacting stateful object with specific structure and behaviour. There are several fundamental concepts of OOD that seem relevant to the UML:

Class and object:

We can think of a class as a blueprint that describes the content of an object. An object is anything which has a state, behaviour and identity that is used to create the instance of a class. The class represents the set of properties or methods the set of properties that are common to all objects of one type.

Class and object in UML

In UML, a class is simply a classifier which describes the set of objects that share the same features, constraints or semantics.

UML 2.0 describes the object as an individual with a state and relationships to other objects where each state of the object can identify the values for the object properties of the object classifier. (Fakhroutdinov, 2019)

Encapsulation

Encapsulation is the fundamental concept in object-oriented design that is used to hide the values or state of structured data object inside a class, preventing unauthorized parties with direct access to them.

Encapsulation in UML

UML specification does not provide any definition of encapsulation; however, it is used loosely in several contexts. In the latest version of UML, the encapsulated classifier is a structured classifier isolated from its environment by using ports. Each port specifies the distinct interaction point between classifier and its environment.

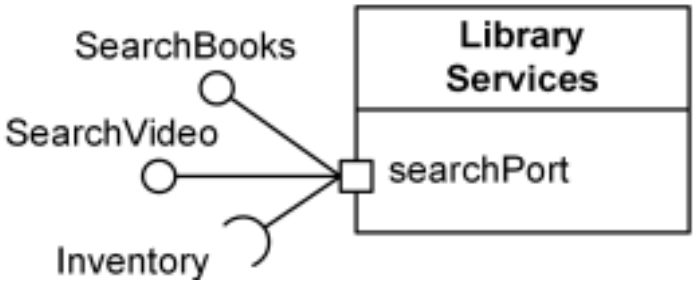

Figure 15 library services is classifier encapsulated through searchPort port (Fakhroutdinov, 2019)

Inheritance

In object-oriented design, inheritance is a technique that new objects take on the properties of existing objects. A child inherits visible properties and methods from its parent and also form or add new properties for itself.

Inheritance in UML

Inheritance, details state that with speculation practising classifier acquires highlights of the broader classifier. Any requirement applying to examples of the general classifier likewise applies to occasions of the classifier.

Abstraction

Abstraction is about removing irrelevant details of implementation to focus on the relevant details.

Abstraction in UML

Abstraction in UML follows as the abstraction of OOD. UML provides different types of abstraction called realization. Realization refers to specialized abstraction between two sets of model elements, specification and implementation.

4.1 Principles of Object-Oriented Programming

Objects are anything which has an identity, state and behaviour. Objects carry out behaviour in response to the message they receive. Methods defined in a class definition, represent the behaviour that instances of that class will have during runtime. This behaviour is invoked by sending messages to that object. The five SOLID principles and techniques of object-oriented programming help in managing module dependencies.

4.1.1 The Open-Closed Principle (OCP)

The open-closed principle states "A module should be open for extension but closed for modification." The module should be written in such a way that they can be extended, without changing the source code of the modules. The new features can be added to existing code without changing it. (Martin, 2000)

4.1.2 The Liskov Substitution Principle (LSP)

“Subclasses should be substitutable for their base class”.

The subclass objects should be able to replace parent class objects without compromising application integrity. Meaning, we should create such derived class objects which can replace objects of the base class without modifying its behaviour. (CodeMaze.com, 2016)

4.1.3 The Dependency Inversion Principle (DIP)

“dependent upon Abstractions. Do not depend upon concretions.”

When higher-level modules depend on rules defined by lower-level modules leads to the violation of the dependency inversion principle. The strategy of this principle is that the modules should depend upon interfaces or abstract functions and classes, rather than upon concrete functions or classes. (Martin, 2000)

4.1.4 The Interface Segregation Principle (ISP)

“Clients should not be forced to depend upon interfaces that they do not use.”

This principle of OOP states that no clients should be forced to depend on methods they do not use. The goal of this principle is to reduce the effects and frequency of required changes by splitting the software into multiple, independent parts. (Martin, 2000)

4.1.5 Single Responsibility Principle (SRP)

“The class should have only one reason to change”

Single responsibility principle states the class should have only one reason to change. In case of requirement changes, only the single place in the application code would require modification. Giving too many responsibilities to a class can be a big violator of single responsibility principle that leads to the fragile and poor code design. (Martin, 2000)

As a part of this project, I need to develop a simple web application using CMS. So I decided to write briefly about the Content Management system and it's typing. I tried to explain about CMS, what technology is it, and how it is helping people for designing attractive websites without the prior knowledge in programming.

5. Content Management System

CMS is a type of web application software that provides all the necessary tools and resources for users to create, modify or manage digital content without the need of the specialized technical experience. CMS provides support to users to create and manage the content for a web application. The CMS can handle all the necessary tools for the user to create content, storing information and manage content according to the user need. The user doesn't need to worry about any programming languages and technical resources. (Kinsta, 2019)

On a technical level, the CMS is made by two distinct parts i.e. Content Management Application, which allows users to easily add, modify or manage digital content in webpages. On the other, Content Delivery Application provides the backend features which accept the content, store it properly and makes it visible to the visitors. The CMS can be divided into two sub-groups:

5.1 Web Content Management System:

WCM are designed to provide capabilities for multiple users at different permission level to create, edit, publish, and report the content such as text, embedded graphics, photo, video, audio and program code of a website. (Kohan, 2019)

5.2 Enterprise Content Management System:

ECMS is an application that supports multiple users manage, store and deliver content in a collaborative environment. It supports multiple functionalities that provides a full-scale Content Management System tailored for a company's organization and processes. (Kohan, 2019)

WordPress

Open source, graphical user interface feature and user-friendly environment make WordPress the most popular CMS. It has the massive content management tools of optional feature plugins and themes of any CMS and can be customized to suit any vision. The features provided by WordPress like free plugins and themes make the users control, store, schedule and publish the digital content in their site. (Bluleadz, 2018)

WordPress functions based on PHP and MYSQL can be utilized as the part of Internet Hosting Services (wordpress.com) or can be run on a local computer as it acts as its web server (wordpress.org). It is one of the most popular web content management software in use mostly for blogging however it has gained its popularity in mailing list and forums, media galleries, learning management system and online stores. It stores the content and provides a user-friendly interface for the user to create, edit and publish web pages requiring nothing beyond the domain and hosting services.

The high use of plugins and themes developed by many people increases the chance of suspicious or malicious code through the site's functionalities. It can lead to a prime target for cybercriminals. (CyberChimps Inc., 2010)

6. Web application for the Tourist visiting Nepal

6.1 Background – Tourism in Nepal

Nepal is a country full of landscapes, mountains, diversity, arts and culture. It is situated in the southern part of Asia between China and India. Nepal is one of Asia's most amazing tourism destination. Its mountains, and rivers provide unlimited opportunities for adventure activities. Tourism is the part of Nepalese culture. It is the backbone for the overall economic development and enhancement of the lifestyle of people. It is the main source of foreign exchange and revenue. the government of Nepal is doing various programs and events to enhance the tourism industry. Recently, Nepal Government has organized "Visit Nepal 2020 A lifetime experience" whose aim is to welcome two million tourists from the world. Different organization perform different activities to make them successful. They organize the cultural program, that highlights the art and culture, language and lifestyle of the people. Sadly, the outbreak of Novel Corona Virus (COVID 19) cause the Nepal government to close all the air flights for a period.

Mountaineering:

The Himalayas are the world's most spectacular mountains, exotic and most adventures destination one can experience, including eight of the world's tallest peaks. Mt. Everest, Kanchenjunga, Lhotse, Makalu, Annapurna etc are some of the tallest peaks above 8000m. Mountaineering can be challenging yet exhilarating experience. Mountaineers can face altitude sickness, bad weather condition and dangerous terrain. It is very important to take permits from the Ministry of Tourism and Aviation for climbing 135 highest peaks, and Nepal mountaineering association is responsible for issuing permits for the 18 expedition peaks. (Simm, 2020)

Trekking:

Nepal's prolific trekking routes with eight of the ten highest summits in the world provide spectacular landscapes and exotic wildlife. Travellers will find many the trekking hub in the street of Kathmandu Pokhara, with guides, organized tours, and gear for sale or rent. The favourable season for trekking is dry and warm seasons: March-June, September-November. Trekking is not necessarily wandering alone through an uncharted wilderness.

Travellers will discover hundreds of locals passing each day as they haul food, water and other necessities to their villages along with other travellers. Tea house and homestay in the village can be organized which allows trekkers opportunity to take rest and recover. Required permits are mandatory as police checkpoints and park officers can check the permits at any time. There will be two or three permits, one will belong conservation area or national park, others will be for Trekkers Information Management System card and the last one is restricted area entry permit. (WikiTravel, 2020)

White-water Rafting:

Nepal is a popular destination for trekkers and mountaineers, but those in-the-know rates it as one of the world's best destination for white water rafting. Karnali, Tamur, Trishuli, Bhote Koshi, Sun Koshi are some of the popular rivers for rafting. The rivers are long and clean and surrounded by mountains and jungle landscapes. Travellers can find the series of world's most outstanding river journey, ranging from a steep, adrift mountain stream to classic and big volume wilderness expedition. September-November is the best time to go for rafting in Nepal. Water level are usually highest, this can make some exciting rapids.

Paragliding:

Paragliding can be the lifetime experience while enjoying the views of snow-capped mountains, pristine lakes and verdant valleys. Travellers can fly in October to March, depending on the weather, in Pokhara valley, near the Annapurna Range. Learn to fly or take a tandem flight with an experienced pilot or even go paragliding accompanied by hawks. (Nepal Tourism Board, 2020)

Canyoning:

Canyoning is a bit unusual water sport but very adventurous for adventure seekers. It is coming down from water canyon by either abseiling, jumping or sliding through canyon walls and waterfalls into deep pools of water below. The refreshing waterfalls, from among the deep gorges, canyoning has become the popular water sport for travellers. (Nepal Tourism Board, 2020)

Jungle Safari:

If you have dreamed of walking in the forest enjoying wildlife, Chitwan National Park and Bardiya National Park provides an adventurous safari package. This includes elephant ride, jeep ride, canoe ride exploring wildlife preservation and vegetation. The cool thing is traveller can enjoy elephant ride exploring the grassland and core area of the park. You can also enjoy bathing with an elephant and take a unique shower with it.

6.2 Impact of Technology in Tourism Industry

It is always complicated to visit new places without any guidelines. No need to worry, today's world is all about technology. The technology has brought a revolution in the travel and tourism industry. They can travel anywhere in the world with ease. People prefer the portable and user-friendly application to plan their destination for holidays. The arrangement with the travel agents is now almost outdated because there are many applications available in the internet market from where one can easily plan their trips. According to a survey by eMarketer, travel-based mobile apps is the seventh most downloaded app category and almost 60% of smartphone users regularly use travel apps while planning trips.

It is no surprise; the development of applications has not only improved the travel experience but also benefited tourism-based companies in a big way. It is a powerful marketing tool for travel companies. All the necessary features, guidebooks, leaflets, and compass have been updated by fully-featured applications that are accessible anytime anywhere. It is easy to reach out millions of users by a simple click and race miles ahead of competitors.

Travellers use the application as a medium for ticketing, hotels booking, getting destination details, weather forecast, locating local destination and food outlets; exploring a new destination, getting reviews about the destination they plan to visit, sharing their travelling experiences. Moreover, travel application provides more services including navigation, security and e-commerce creating high potential of monetization and revenue generation.

6.3 SWOT Analysis

- **Strength:**

  For a small country like Nepal, Tourism has played a tremendous role in economic growth and employment. People not only depend on agriculture, but they are maintaining their lifestyle through tourism. Besides, there are also social and cultural advantages of tourism. It is a great opportunity for local communities to look at their history and cultural heritage and develop their community identity.

- **Weakness:**

  Although there are many opportunities for growth through tourism the effective policies, geographical condition, political instability has caused hindrance in the tourism industry.

- **Opportunity:**

  It is a great opportunity for the entrepreneurs to establish new services and products, or facilities that would not be sustainable based on the local population of residents alone. Tourists are the potential costumers, and the right business strategy allows for fantastic success.

- **Threat:**

  The outbreak of the Novel Corona Virus has become a serious threat to the tourism industry. It has harmed the country's economy. It has reduced the significant amount in the GDP.

7. Practical Part

7.1 The web application

This web application is designed for travellers visiting Nepal. It provides basic information for travellers who are planning to visit Nepal about places, culture and tradition, people and language, and accommodation. Travellers can visit the site and plan the trip to their interest.

7.2 Functional Requirements

Functional requirements specify particular results of a system. The implementation of functional requirements is detailed the communication among the application system and the users regardless of its execution. The following are the main functional requirements of this web application:

- Functionality to register for new users and login for existing users.
- Enable booking and execute booking and be able to make a payment
- Users can enquire using a form.
- The system should provide easy navigation and visualization.
- The system can authenticate and authorize different users and their assigned privileges.
- The privileges to change the password for the users.
- The users can manage their dashboard and send feedback
- The users can arrange the date, several people to book a tour.

7.3 Non-Functional Requirements

Non-functional requirement of a system are the quality attributes of a system, which helps to verify the quality attribute available in the system. It is understood that the failing to meet non-functional requirement of a system can result that fail to meet the user requirement and fail to satisfy user needs. The software system is based on responsiveness, usability, security, portability, and other non-functional standards that are critical to the success of the software system.

- The application shall allow several booking to be made at the same time.
- The application shall be easy to use by all the users and admin.
- It should guarantee user satisfaction.
- The application should respond in the minimum time without crashing.
- It should only provide a valid result.

7.4 Analysis Model

The necessary analysis of system design for this application is discussed using different UML diagrams for the visualization. The class diagram is used to show the static behaviour of a system and interactions between the entities of the system are described using the dynamic models of the UML.

7.5 Data Dictionary

- **Customer:** the person who uses the application to see the available services and can book and buy the listed products.
- **System admin:** the person responsible to manage all the backend data and information and perform all the clerical role related to orders of customers.
- **Services:** a list of available destination that interest the customer. it shows the basic information about the destination, maximum number of people who can visit, price and ability for feedback.
- **Booking:** feature of an application for the successful purchase of the services from the customers.

7.6 Static model

Class Diagram:

The following figure shows the overview of a class diagram. It depicts the static relationships between objects presents in the application. Each class represents the objects with their related attributes and methods/operation. Before building the diagram it is necessary to recognize relationships between the classes, their communication, and their links, associations, generalizations with other classes in the diagram. As shown in the figure below, the classes and their relationships are designed considering the project description of web application. This includes the classes of person, system admin, customer, services, book and payment. The relationship among the classes and their respective multiplicities are identified in the diagrams.

We can see on the figure no, the class person inherits or generalizes two other classes i.e. Admin and Customer. These two classes inherit all the attributes and method from the parent Person class and also has their separate methods. The parent class Person abstracts the common attributes from the child classes. Even though the child classes have their own attribute, in this case we don't see the unique attribute presents in the child classes.

The other type of relationships we see here is the composition association between the Services class and Booking class. This type of association shows the strong relationships between the class as it implies the ownership from the whole to the part. In this case the services class has the composition relationships with Booking class. Destroying the services class leads to the destroying the Booking class. Booking class simply can not exist without the Service class.

In the following class diagram, class Customer to Booking and class Booking to Payment shared a connection which can be represents as the association relationships. There is an indication of multiplicity of an association by adding multiplicity adornments to the line denoting the association.

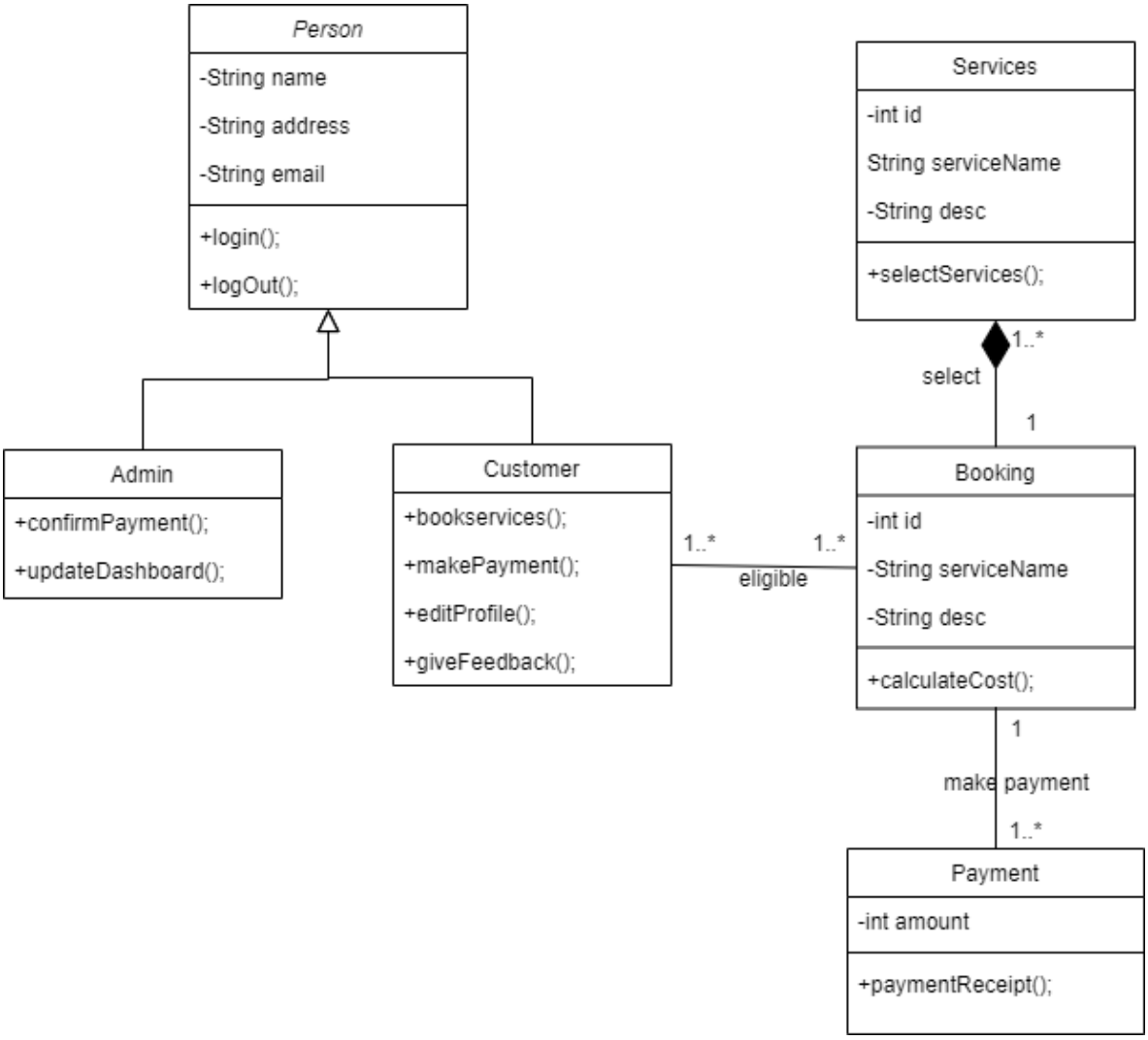

Figure 16 class diagram and their relationships

7.7 Dynamic model:

Use case diagram:

The following figure for this web application shows the use of cases involving customer and system admin as an actor. As the customer first visit the web application, they are prompted to log in to be able to book the available services. The login use case is an include relationship which shows the dependency with verify password use case. Every time the login use case is implemented the verify password use case is implemented as well. On the other hand, the

extended use case is not necessarily implemented whenever the login use case is implemented. The users are available to see the available destination and can book the tour. The book destination use case shows the dependency relationship on contact details and payment use case. The user can complete the booking when contact details and payment are successfully made. The system admin is another actor in the following use case diagram. It has the responsibility to check if the user payment is confirmed.

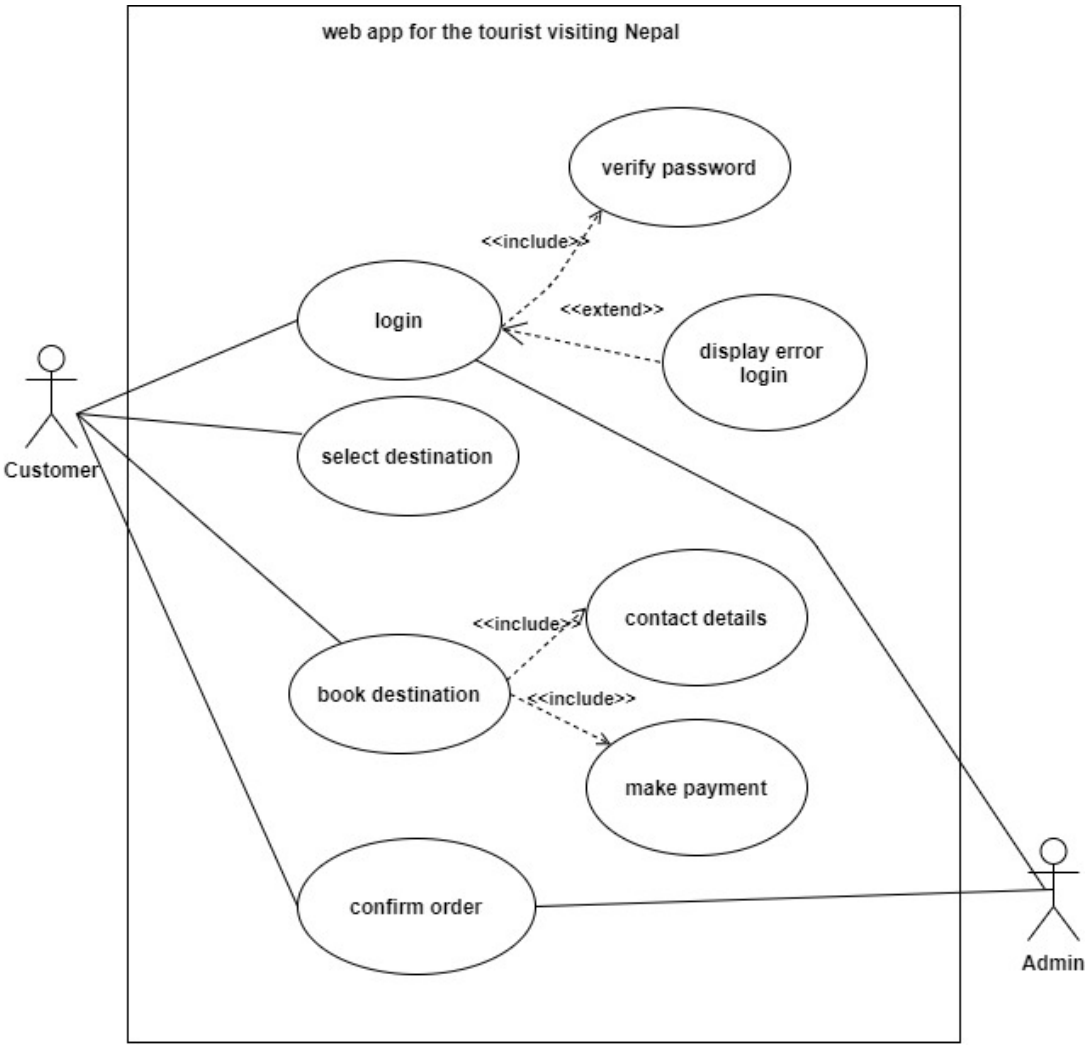

Figure 17 use case diagram for a web application (author, 2020)

Sequence diagram:

The sequence diagrams are the type of UML diagrams that shows how the objects in the system interact with each other. These diagrams show interactions in the order that takes place. In another word, they show the sequence of events.

The figure shows the implementation of the sequence diagram. The process starts as the user visits the site and register to use the available services. The scenario starts from the user attempts to register by filling the forms and stored in the database. When the registration is complete, the user can log in using the username and password. If the login credential provided by the user is authenticated and authorized, the user can see and interact with the available services in the system.

Sequence diagram for booking destinations

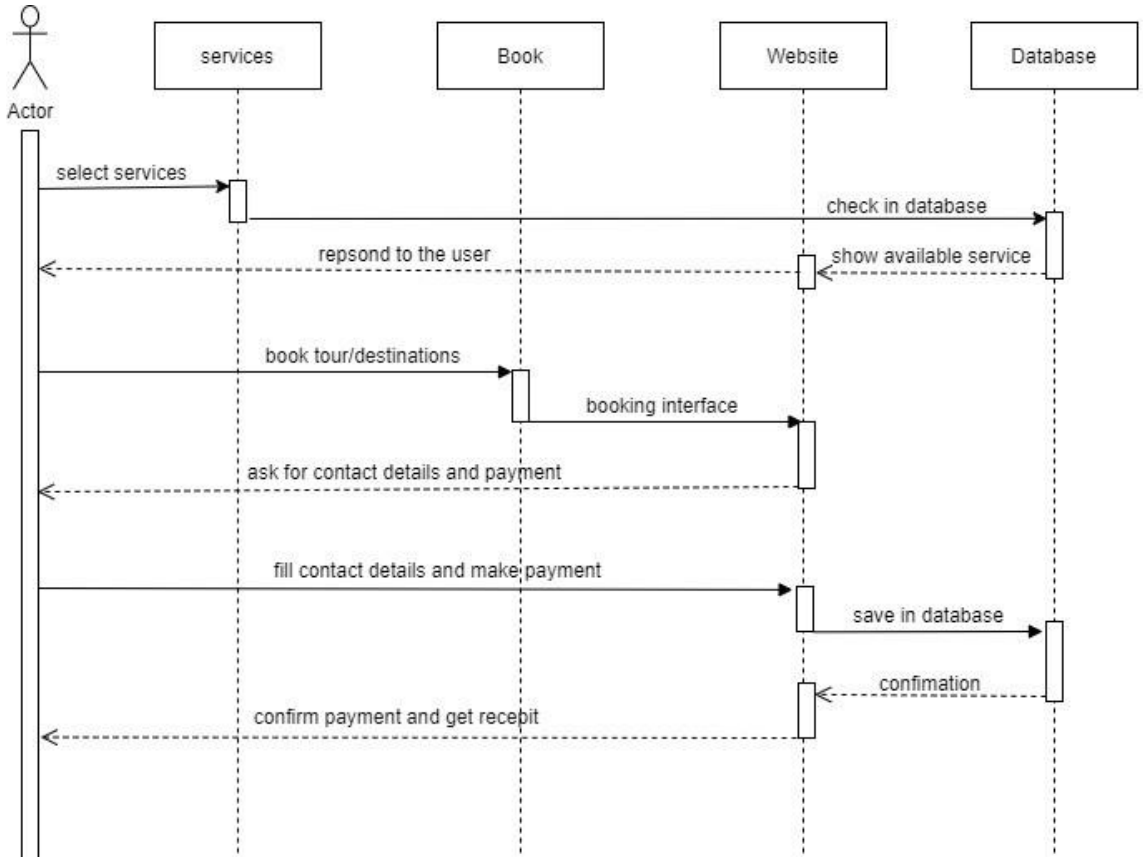

Figure 18 Sequence diagram for booking services (author, 2020)

Sequence diagram for registration

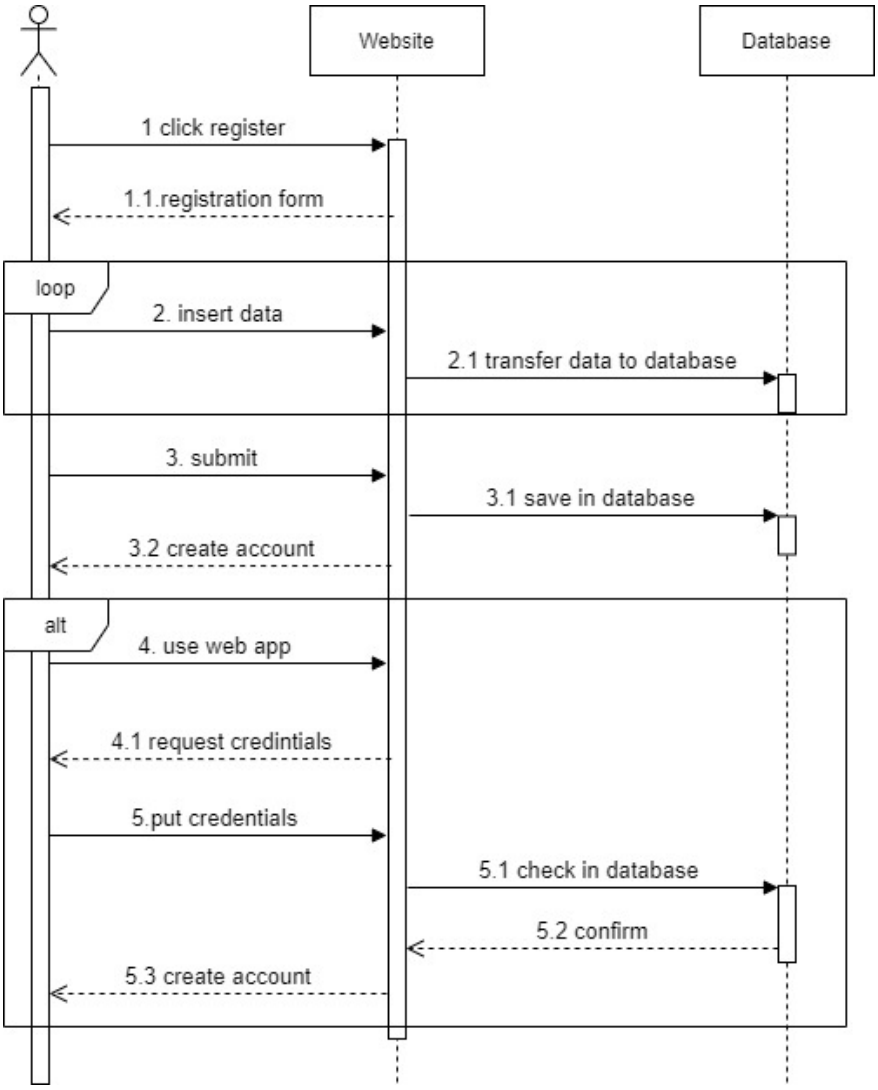

Figure 19 sequence diagram for register new user (author, 2020)

Activity diagram

Activity diagram of the booking procedure

The figure 21 shows the details procedure of booking of a tour in a web app. The users can see the available tours in the system. The web system provided detail information to the users. If the user is interested in booking, they can proceed to book providing the contact details and making a payment and received the confirmation.

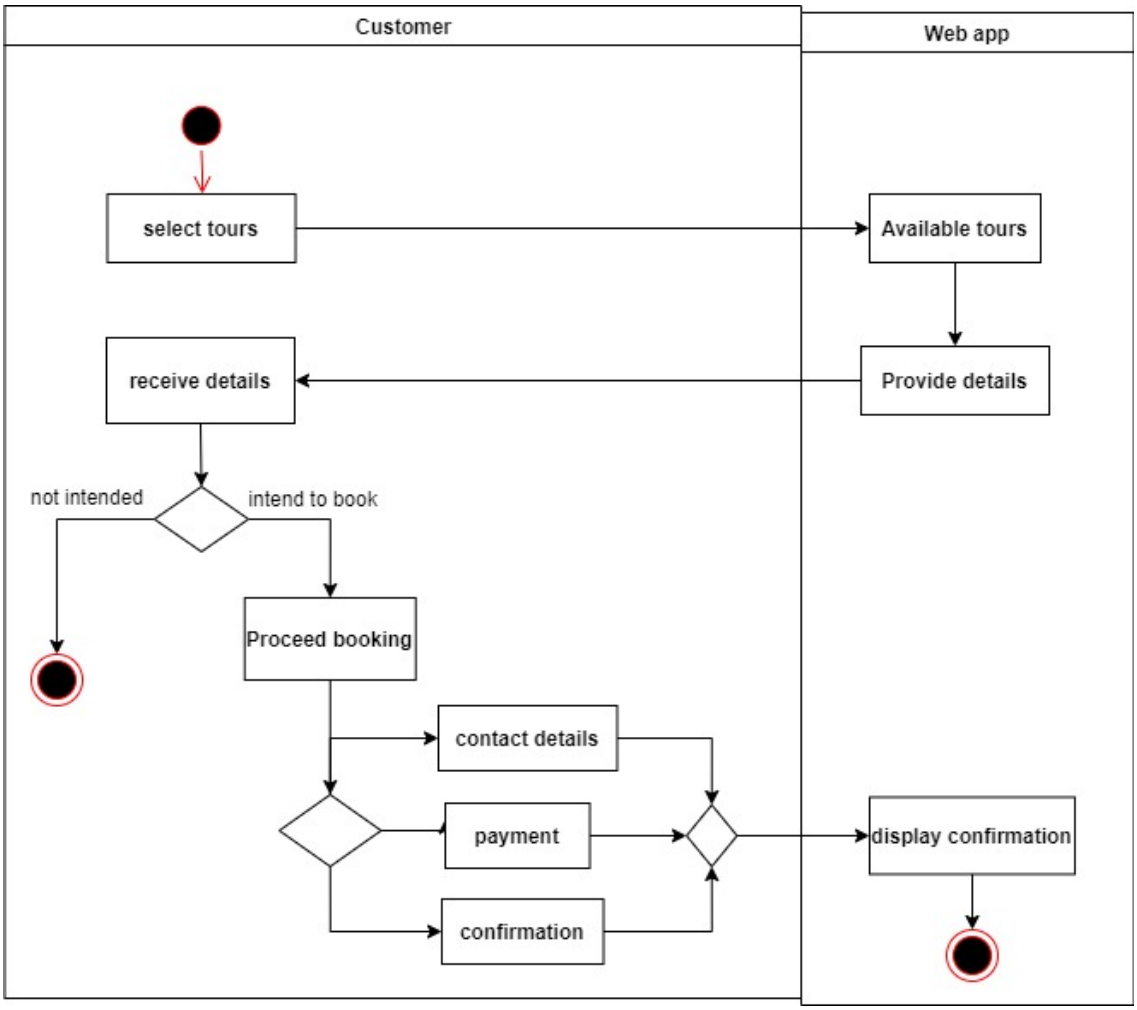

Figure 20 activity diagram for booking services in a web application (author, 2020)

Activity diagram for login procedure

The following figure shows the login procedure for the users. The registered users should have username and login to access their profile. The system will verify the user using if else condition. If the conditions are met, the users are authorised to access the web app and if not it prompt the error message.

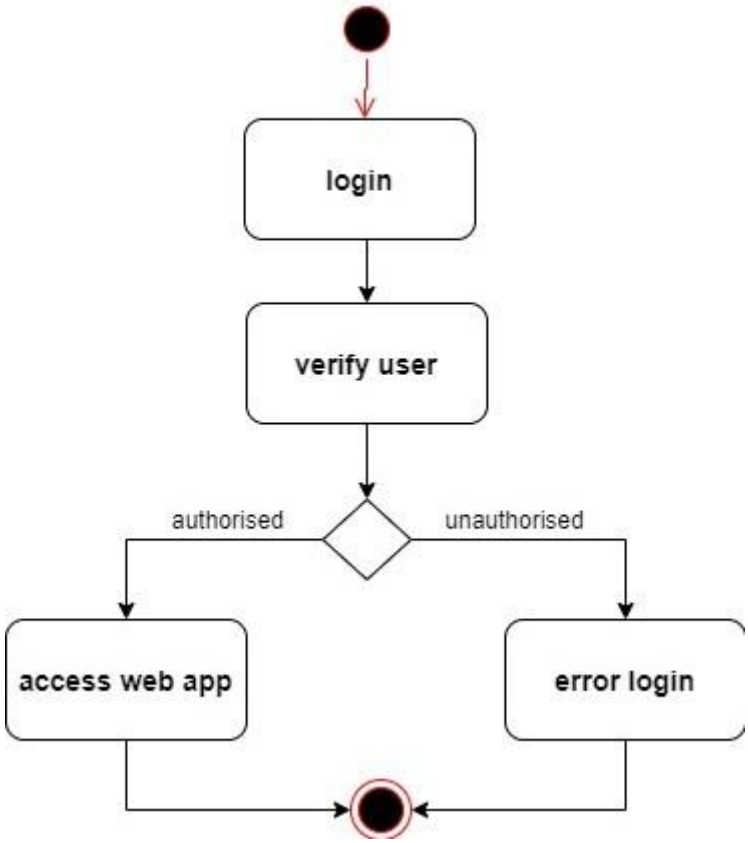

Figure 21 activity diagram for login procedure (author, 2020)

8. Architectural Design

While designing the web application process, it is necessary to follow standard architectural designing procedures for better performance, accuracy, and time efficiency. It is widely

followed and accepted by system development community. The following criterion for architectural design are discussed below:

Performance:

This criterion responses to the fast response. The more quickly the web app response the more the customer are satisfied. Navigating app, information retrieval and fast response can enhance the performance of an application.

Reliability:

This ensures there is always the backup if there is any fault or failures in the system. Failures are uncertain, it can happen any time. So, the system should always be ready to tackle these problems. It must be eligible to recover in less time possible.

Availability:

To ensure the 24/7 availability of web application the host with high percentage of SLA should be used that ultimately make the recovery time quicker with a very small down time when the system fails or crashes.

Maintenance:

It is necessary to keep the system maintained to maximize the performance and system functionality for the overall satisfaction. It should be in the stage of easy maintainability.

Security:

Security is vital for any software application. The system software, hardware and users must be secured using multiple security procedures. Users should be managed to have access based on their needs. The system should have the ability to detect suspicious activities like several login attempted failures, payment failures etc.

End user:

At last, the main goal is the user satisfaction. For that easy functionalities, navigations, menus, user friendly interface should be used.

Following these procedures enhance the overall performance of a system and guarantee the user satisfaction. It ensures the system in maintained and highly functional and provides securities to any threads.

9. Implementation of web application

The prototype of the web application for tourist visiting Nepal is designed and implemented using a content management system i.e. WordPress, which is a great tool to design, implement, and maintain the website. WordPress functions based on PHP and MYSQL can be utilized as the part of Internet Hosting Services (wordpress.com) or can be run on a local computer as it acts as its web server (wordpress.org). It is one of the most popular web content management software in use mostly for blogging however it has gained its popularity in mailing list and forums, media galleries, learning management system and online stores. It stores the content and provides a user-friendly interface for the user to create, edit and publish web pages requiring nothing beyond the domain and hosting services. The UML is used in designing the prototype of an application which enables different stakeholders to have the same view of the system in the development process. It is the standard for visualizing the architectural design of a system in a diagram. Furthermore, the UML reminds the software developers and architect to design a prototype for proper planning, design and whether it meets the requirements. The UML provides software developers, system architect and software engineers with necessary tools for analysis, design and implementation of the software-based system as well as modelling the business and similar process.

The development process of a web application for tourism services is dependent on the source codes. The user friendly interface helps the users to easily navigate the system, use the functionalities of the system and be able to read information with the right font and colours. Some screenshots of the design in web application in WordPress are shown below.

The following figures depict the overview of an application designed and implemented using WordPress. It shows the main page of an application as a user enters the URL. It has the list of features and the user can click and view the information of their interest. The user can log in or signup and learn the details about the services including price, booking and enquiry.

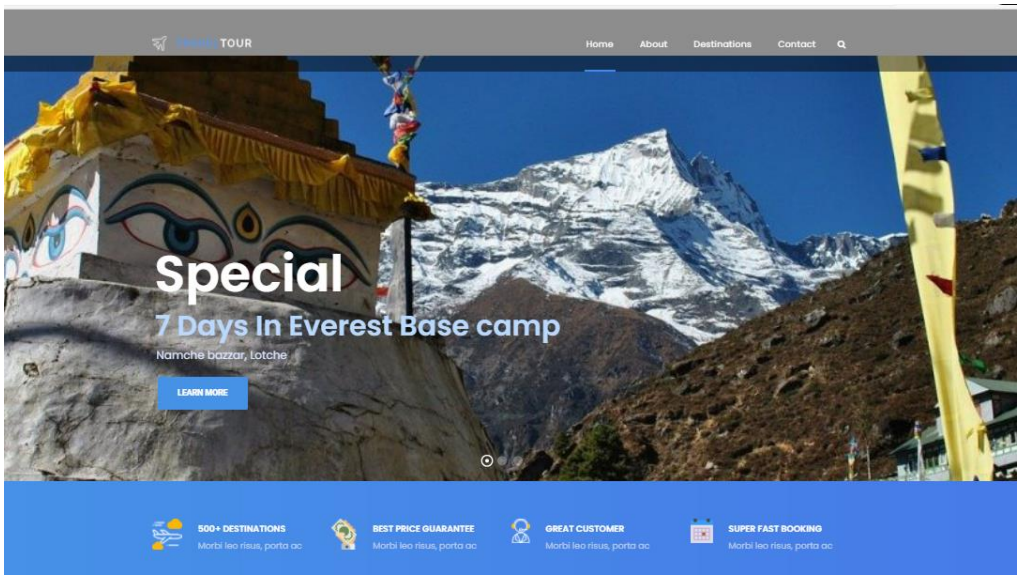

Figure 22 home page for a web app

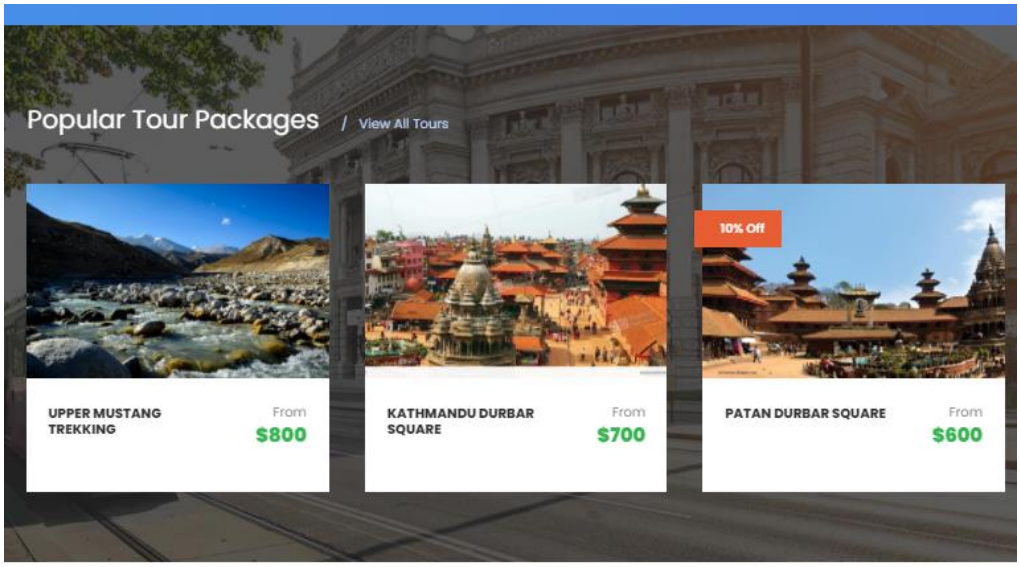

Figure 23 available services in home page

The web application services for tourist visiting Nepal has different navigation menus which are linked with related functionalities and services of the web site. The primary menus i.e. Home, About, Destination, contact and login are discussed as follows:

Home: The home page is the main page users see when they enter the URL. Any users who can log in or who are just a visitor see the home page. The home page and an introductory well come page for all.

About: The about page provides all the necessary information about the company running this website. It depicts the background history of the company briefly. Users can see the company's visions and objectives and the services it provides to the public

Register: the users can fill their details in the register page to create an account. The user can either register or just visit the site. The registered users and their details must be saved in the database. Certain functionalities like booking the tour is only possible when the users are registered and logged in in the system.

Login: this is the page where the users can login using their email and password. The features to manage dashboard, edit profile, booking and make payment are only possible after the user logged in. On the other hand, the system administrators will have a full right to access and have control over the system.

The figure shows an overview of the Login and Register Pages. the user can create a new account if they wish to buy the product and see the available services provided by the application.

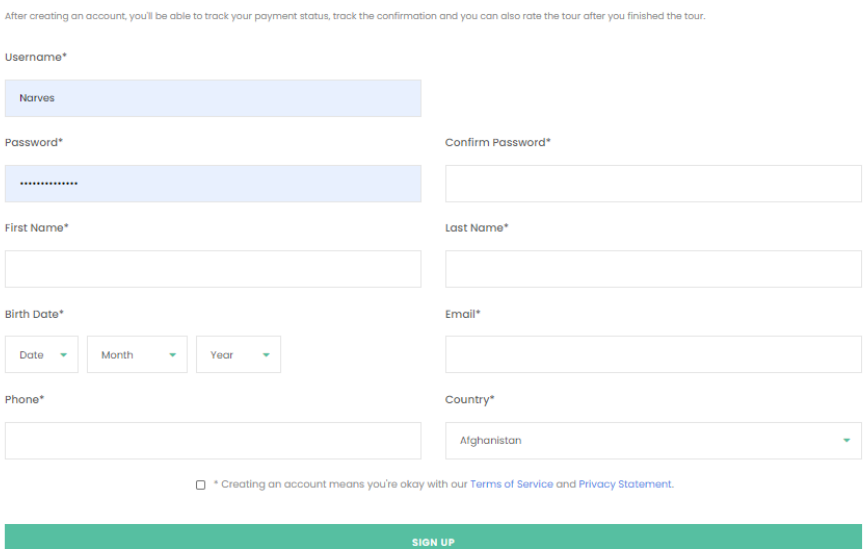

Figure 24 registration form for web application

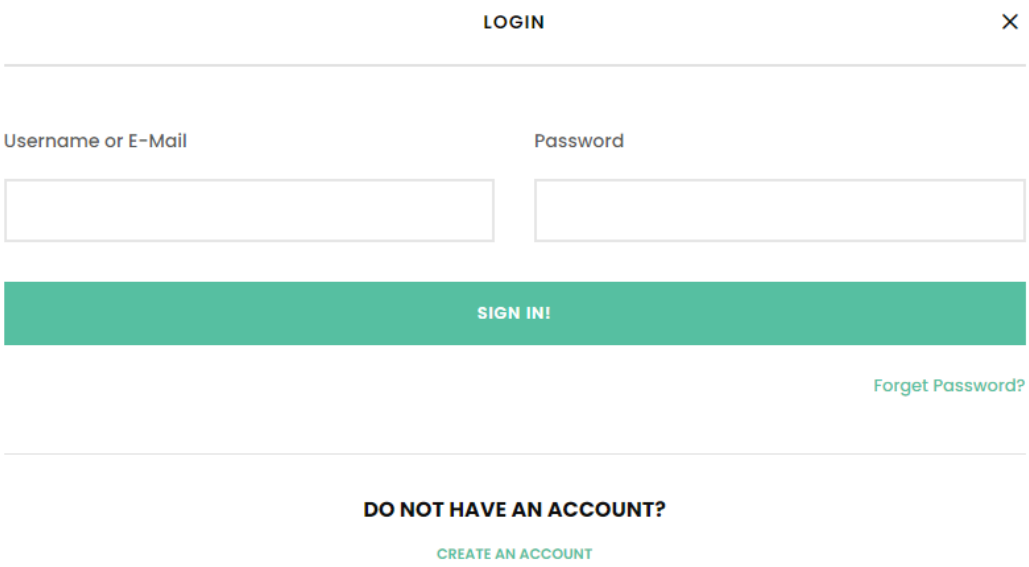

Figure 25 login form for web application (author, 2020)

The main objective of this project is to provide the necessary services to the travellers and give details information of their interest. So, the service page designed in WordPress contains the tour details, prices and other services.

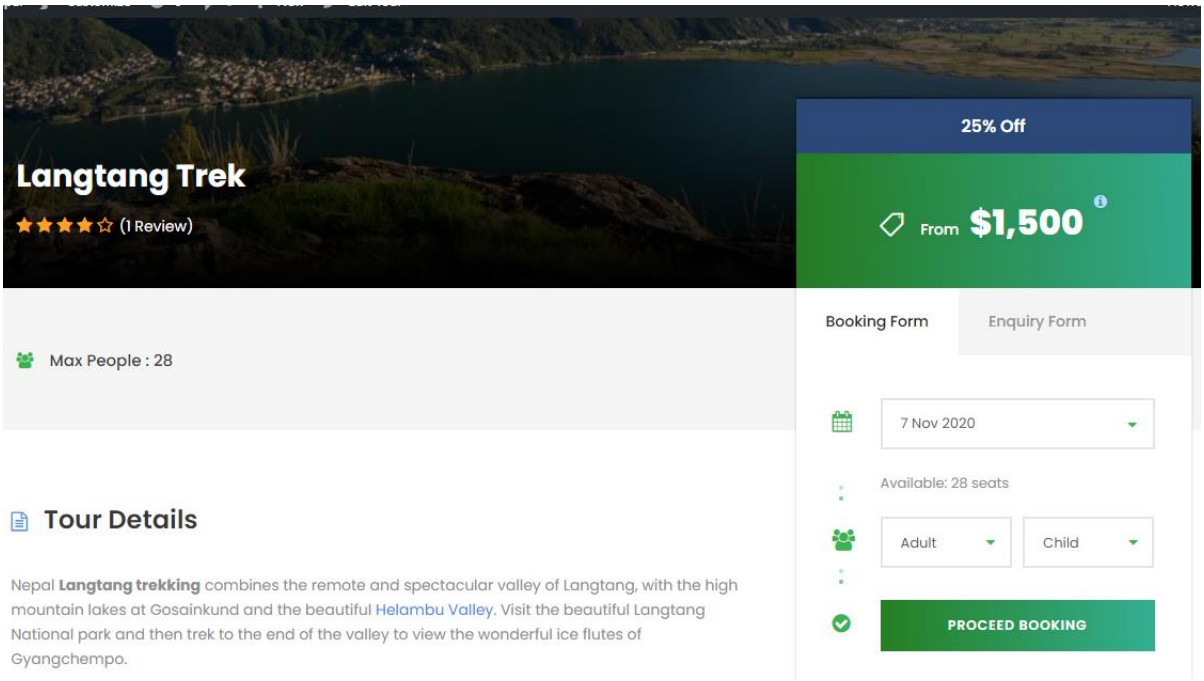

Figure 26 service page with tour details and booking procedure (author, 2020)

Once the traveller is interested in a particular tour, they can proceed for booking. They can simply fill up the booking form including date, number of people and proceed for next step to authenticate the contact details. The user has to update all the necessary information to proceed to make a payment. The payment can be made using visa or master card fully or partially or the user can book the tour and make a payment later. The following figures below show the method of booking, contact details and a successful payment for a particular service:

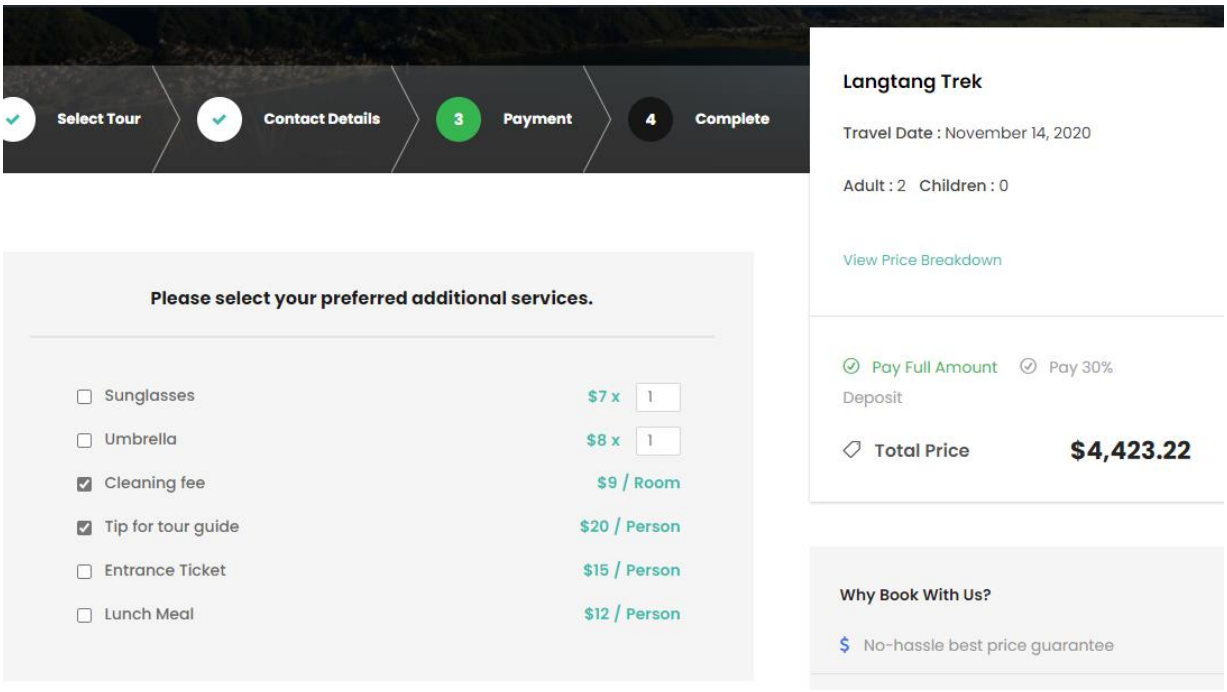

Figure 27 payment process and additional services in a web app (author, 2020)

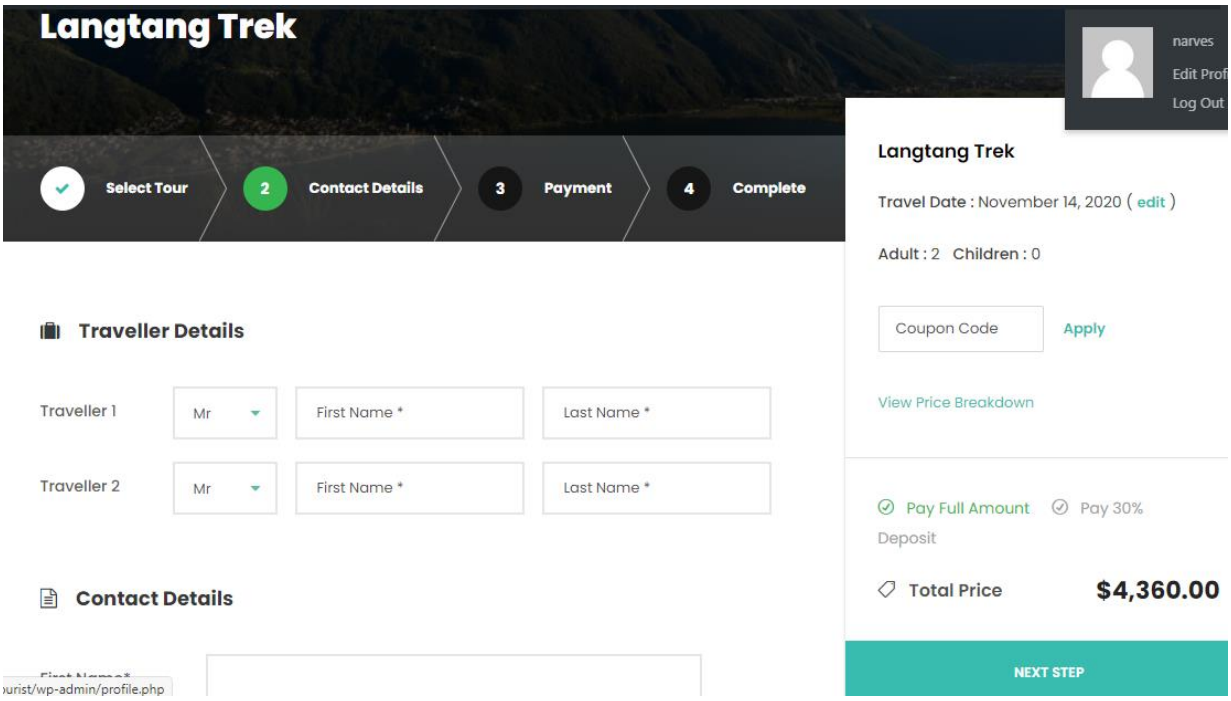

Figure 28 contact details form after booking services (author, 2020)

10. Conclusion

This dissertation demonstrates designing a prototype of a web application for the tourist visiting Nepal. The purpose of this application design is to provide detail information about the destination listed in the application. The users can view the detailed overview of the places, cultural events, geographic and weather conditions. The system is implemented in UML which shows the relationship between different elements and their communication with each other. The UML has provided a strong foundation in designing a prototype for proper planning, analysis, design, and implementation of the system before writing the code. After the completion of designing a prototype of an application a content management system, i.e. WordPress is used to design the user interface of a system. The system has two access levels: the first level without the need to login and pays anything will be for providing basic information. The second level has personalised to an individual user and track their services and provide some feedback.

Following the objectives, the third chapter performed the literature review on an information system, Unified Modelling Language and its history and software development lifecycle. These topics are essential for designing the web information system for the tourist visiting Nepal. UML is used to design and implement a prototype showing the relationship between different elements in the system. On the other hand, the strategy of the software development lifecycle is followed to meet the required objectives.

The system implementation is performed in the practical part of the web application system. The unified modelling language is used to design the system which shows the dynamic and static behaviour of the system. The class diagram is used to show the static behaviour of the system. It describes the main objects in the system and their relationship. Furthermore, use case diagram, sequence diagram and activity diagram are used to show the interactions of the objects in the system. Following the prototype designed in UML, some major functionalities of the system and a user interface is built using WordPress.

Based on the result of the study, this web application for tourist visiting Nepal provides the necessary benefits for the users with the available services listed in the app. It has the potential to be successful in the future.

References

Bizzdesign. (2017). */display/knowledge/Available+UML+diagrams+and+predefined+primitive+types*. Retrieved from Available UML diagrams and predefined primitive types..: https://support.bizzdesign.com

Bluleadz. (2018). *bluleadz*. Retrieved from the-8-best-marketing-cms-platforms-in-2018: http://www.bluleadz.com

Bourgeois, D. T. (2014). *Information System for Business and Beyond.* The Saylor Academy. Retrieved from https:/www.courses.lumenlearning.com

Burback, R. L. (1998). */~burback/watersluice/node56.html*. Retrieved from Object modelling Technique: http://infolab.stanford.edu

CodeMaze.com. (2016). *SOLID PRINCIPLE IN C# - Liskov Substitution Principle- code maze*. Retrieved from code-maze.com/liskov-substitution-principle: https://www.code-maze.com

CyberChimps Inc. (2010). *pros-and-cons-of-wordpress*. Retrieved from https://cyberchimps.com

EDUCBA. (2019). *uml-deployment-diagram*. Retrieved from UML deployment diagram: https://www.educba.com

Fakhroutdinov, K. (2019). Retrieved from Unified Modelling Lanagage, description, uml diagram examples...: https://www.uml-diagrams.org/

Fowler, M. (2000). In M. Fowler, *UML Distilled: a brief guide to standard object modelling language* (p. 23). Reading, Massachusetts, England: Addison Wesley Longman, Inc. Reading, Massachusetts.

Fowler, M. (2003). UML Distilled 3rd ediion. In M. Fowler, *UML Distilled 3rd edition* (p. 68). Addison-Wesley.

Fowler, M. (2003). *UML Distilled, 3rd Edition.* Addison-Wesely.

IBM. (2015). */support/knowledgecenter/SSCLKU_7.5.5/com.ibm.xtools.modeler.doc/topics/cassnclss.html*. Retrieved from association class: https://www.ibm.com

Jacobson, I. (2013). OOSE Model. In I. Jacobson, *Giants of Computing* (p. 155). Gerard O'Regan.

Kinsta. (2019). *content-management-system*. Retrieved from Kinsta: https://kinsta.com/knowledgebase/content-management-system/

Kohan, B. (2019). *what-is-cms-content-management-system.html*. Retrieved from comentum: www.comentum.com

Martin, R. C. (2000). *Design Principles and Design Patterns.* Retrieved from objectMentor: http://www.objectmentor.com

Nepal Tourism Board. (2020). *things-to-do/canyoning.html*. Retrieved from Welcome to Nepal |Nepal's official travel and tourist information website: https://www.welcomenepal.com

Object Management Group. (2017, December). */spec/UML/2.5.1/PDF*. Retrieved from unified modelling language: https://www.omg.org

Osio Labs. (2019). *what-is-drupal*. Retrieved from drupalize.me: https://drupalize.me/

Simm, C. (2020). *adventure-tourism-nepal-56272.html*. Retrieved from adventure tourism in Nepal: https://traveltips.usatoday.com/

softwareTestingHelp. (2017). */software-development-life-cycle-sdlc/*. Retrieved from what is SDLC(software development lifecycle) phases: https://www.softwaretestinghelp.com/

sourcemaking. (2019). *history of UML: methods and notation*. Retrieved from /uml/basic-principles-and-background/history-of-uml-methods-and-notations: https://sourcemaking.com

SparxSystem. (2000). *enterprise_architect_user_guide/14.0/model_domains/classdiagram.html*. Retrieved from class diagram/ enterprise architecht user guide: https://sparxsystems.com/

SparxSystem. (2007). /downloads/whitepapers/UML_Tutorial_Part_2_Introduction.pdf. *Using UML - Behavioural Modelling Diagrams*, 5. Retrieved from https://sparxsystems.com.au

Technopedia. (2019). */definition/3243/unified-modeling-language-uml*. Retrieved from https://www.techopedia.com

Upwork Global Inc. (2019). *wordpress-vs-squarespace/*. Retrieved from wordpress or squarespace: https://www.upwork.com/

VisualParadigm. (2017). *UML practical guide*. Retrieved from visual paradigm: https://www.visual-paradigm.com

wikipedia. (2019). */wiki/Unified_Modeling_Language*. Retrieved from Unified Modelling Language: https://en.wikipedia.org

WikiTravel. (2020). */en/Trekking_in_Nepal*. Retrieved from Trekking in Nepal - WikiTravel: https://wikitravel.org