**<E-Dukaan>**

**(Project Proposal)**

**Project Code**

<Project code assigned by the Project Office>

**Project Advisor**

Prof Saad Razaq

**Project Manager**

<Name of the project manager>

**Project Team**

**BCSF18E008 Ali Asghar Leader**

**BCSF18E028 M. Saad Saleem Member**

**BCSF18E005 Asif Mehmood Member**

**Submission Date**

1st November 2021

**Table of Contents**

1- Abstract

2- Background and Justification

3- Project Methodology

4- Project Scope

5- High Level Project Plan

**SRS**

[1.](#_heading=h.1ci93xb) Introduction 4

[*1.1*](#_heading=h.3whwml4) *Purpose of Document 4*

[*1.2*](#_heading=h.2bn6wsx) *Project Overview 4*

[*1.3*](#_heading=h.qsh70q) *Scope 4*

[2.](#_heading=h.3as4poj) Overall System Description 4

[*2.1*](#_heading=h.1pxezwc) *User characteristics 4*

[*2.2*](#_heading=h.49x2ik5) *Operating environment 4*

[*2.3*](#_heading=h.2p2csry) *System constraints 4*

[3.](#_heading=h.147n2zr) External Interface Requirements 5

[*3.1*](#_heading=h.3o7alnk) *Hardware Interfaces 5*

[*3.2*](#_heading=h.23ckvvd) *Software Interfaces 5*

[4.](#_heading=h.ihv636) Functional Requirements 5

[5.](#_heading=h.32hioqz) Non-functional Requirements 5

[*5.1*](#_heading=h.1hmsyys) *Performance Requirements 5*

[*5.2*](#_heading=h.41mghml) *Safety Requirements 5*

[*5.3*](#_heading=h.2grqrue) *Security Requirements 5*

[*5.4*](#_heading=h.vx1227) *User Documentation 6*

[6.](#_heading=h.3fwokq0)  6

**SDS**

**Cocommo estimation model**

**References**

1. **Abstract**

We will develop the E-Commerce Site. The objective of developing an E-commerce site is to provide opportunities to the public for buying online products. It will help them to have a record of all the products they want to buy in a single card. Also they can search the available products and check their price. Our site will solve the issue of traffic in the market as if most of the people buy their products online so this will decrease the traffic in the market and also solve the environmental issue.

**2. Background and Justification**  
E-Commerce means buying and selling of products or services over electronic systems such as the Internet and other computer networks. These online shopping sites like **Daraz.pk** and **priceoye.pk** has several problems. For example, unclear customer services policy, Overwhelming competition, Scarcity of trained staff. The proposed project will overcome all of these problems by providing clear customer services. All of the services will be done under the supervision of trained staff.

**3**. **Project Methodology**

Developer will create a layout of an E-commerce site with the help of HTML and CSS. Products will be displayed on E-commerce site. The developer will provide a card to the customer so that he can have a record of all the product they want to buy.

**4. Project Scope**

It will provide the customer with the opportunity to calculate Price and a search column. It will help the people to buy online things by sitting at their own homes. It will be a great advantage to the people in this critical period of Pandemic. It will also provide people with the opportunity that they can buy the genuine products as are shown on the site.

**5. High level Project Plan**

The two important functions will be used in the proposed project that will calculate Price and search. We will take about 4 weeks to complete this activity.

**SRS**

1. **Introduction**

The Software Requirements Specification is designed to document and describe the agreement between the customer and the developer regarding the specification of the software product requested. Its primary purpose is to provide a clear and descriptive “statement of user requirements” that can be used as a reference in further development of the software system. This document is broken into a number of sections used to logically separate the software requirements into easily referenced parts. This Software Requirements Specification aims to describe the Functionality, External Interfaces, Attributes and Design Constraints imposed on Implementation of the software system described throughout the rest of the document. Throughout the description of the software system, the language and terminology used should unambiguous and consistent throughout the document.

* 1. **Purpose of Document**

*Main purpose of this document is to show the real functionalities and problem-solving techniques which are to be done in our project and also show the scope and properties and key points of our project.*

* 1. **Project Overview**

Defining and describing the functions and specifications of the E-Commerce

System is the primary goal of this Software Requirements Specification (SRS).

This Software Requirements Specification illustrates, in clear terms, the system’s primary

Uses and required functionality as specified by our customer.

* 1. **Scope**

*• The Scope of this project is very broad in terms of buying online from home.*

*• It can be used by student in home for self-study guide to buy books or some useful materials.*

*• It can be used for corporate world person who had less time in real life, so they can*

*Buy online.*

*• It can easily understandable by every person because it’s content very •*

*Understandable by a normal person, there is no requirement for elaboration.*

*• It is based on web application, we can easily access it from anywhere.*

1. **Overall System Description**
   1. **User characteristics**

*The Online Shopping system (OSS) application enables vendors to set up online*

*Shops, customers to browse through the shops, and a system administrator to*

*Approve and reject requests for new shops and maintain lists of shop categories.*

*Also, the developer is designing an online shopping site to manage the items in the*

*Shop and also help customers purchase them online without having to visit the shop*

*Physically. The online shopping system will use the internet as the sole method for*

*Selling goods to its consumers.*

* 1. **Operating environment**

*This system provides an easy to solution customer’s to buy the product without*

*Go to the shop and also shop owner to sale the product.*

* 1. **System constraints**

*These are the following Constraints in our project:*

* *Needs Browser or any kind of web search engine to open the site*
* *Needs a good speed internet connection*
* *It is in English language.*
* *Purchases will be only in Pakistan.*
* *Browser needs to be updated to the latest version.*
* *Website is developed for the online buyers only.*
* *HTML, CSS and JavaScript and PHP are used in this project, so we have to manage accordingly.*

1. **External Interface Requirements**

*External Interface Requirements depends on the hardware and software interfaces of the project.*

* 1. **Hardware Interfaces**

*The System must run over the internet, all the hardware shall require to connect*

*Internet will be a hardware interface for the system. As for e.g. Modem, WAN –*

*LAN, Ethernet Cross-Cable.*

* 1. **Software Interfaces**

*The system is on server, so it requires any scripting language like PHP,*

*VBScript etc. The system require Data Base also for the store any transaction of*

*The system like MySQL etc. system also require DNS (domain name space) for the*

*Naming on the internet. At the last user need web browser for interact with the*

*System*

1. **Functional Requirements**

This section provides a requirement overview of the system. Various functional

Modules that can be implemented by the system will be -

**Description**

**Registration**

If customer wants to buy the product then he/she must be registered, unregistered

User can’t go to the shopping cart.

**Login**

Customer logins to the system by entering valid user ID and password for the

Shopping.

**Changes to Cart**

Changes to cart means the customer after login or registration can make order or

Cancel order of the product from the shopping cart.

**Payment**

For customer, there are many type of secure billing will be prepaid as debit or credit

Card, postpaid as after shipping, check or bank draft.

**Logout**

After the payment or surf the product, the customer will log out.

**Search**

There are some search techniques available in which user can search the items.

**Filter**

Filter option available for the users to get more about the product's details and availability.

**Report Generation**

After all transaction, the system can generate the portable document file (.pdf) and

Then sent one copy to the customer’s Email-address and another one for the system

Database to calculate the monthly transaction.

**Technical Issues**

This system will work on client-server architecture. It will require an internet

Server and which will be able to run PHP application. The system should support

Some commonly used browser such as IE etc.

**Interface Requirement**

Various interfaces for the product could be-

1. Login Page

2. Registration Form

3. There will be a screen displaying information about the product that the shop

Having.

4. If the customers select the buy button then another screen of shopping cart will

Be opened.

5. After all transaction the system makes the selling report as portable document

File (.pdf) and sent to the customer E-mail address

1. **Non-functional Requirements**

* 1. **Performance Requirements**

*There is no performance requirement in this system because of the server request.*

*And response is depended on the end user internet connection.*

* 1. **Safety Requirements**

*The system must automatically log out all customers after a period of inactivity.*

*Sensitive data will be encrypted before being sent over insecure connections like*

*The internet.**The reliability of the overall program depends on the reliability of the separate*

*Components. The main pillar of reliability of the system is the backup of the*

*Database, which is continuously maintained and updated to reflect the most recent*

*Changes.*

*Thus, the overall stability of the system depends on the stability of the container and its*

*Underlying operating system*

* 1. **Security Requirements**

*The system use SSL (secured socket layer) in all transactions that include any*

*Confidential customer information.*

*The system must automatically log out all customers after a period of inactivity.*

*The system should not leave any cookies on the customer’s computer containing*

*The user’s password.*

*The system’s back-end servers shall only be accessible to authenticated*

*Administrators.*

*Sensitive data will be encrypted before being sent over insecure connections like*

*The internet.*

* 1. **User Documentation**

*This document is meant to delineate the features of OSS, to serve as a guide*

*To the developers on one hand and a software validation document for the*

*Prospective client on the other. The Online Shopping System (OSS) for*

*Shop web application is intended to provide complete solutions for vendors as well*

*As customers through a single get way using the internet. It will enable vendors to*

*Setup online shops, customer to browse through the shop and purchase them online*

*Without having to visit the shop physically. The administration module will enable*

*A system administrator to approve and reject requests for new shops and maintain*

*Various lists of shop category.*

***Functional Specifications***

* **Introduction**
* **Purpose of Document**

We will develop the E-Dukaan Site. The objective of developing an E-Dukaan site is to provide opportunities to the public for buying online products. It will help them to have a record of all the products they want to buy in a single card. Also they can search the available products and check their price. Our site will solve the issue of traffic in the market as if most of the people buy their products online so this will decrease the traffic in the market and also solve the environmental issue.

* **Project Overview**

E-Dukaan means buying and selling of products or services over electronic systems such as the Internet and other computer networks. These online shopping sites like **Daraz.pk** and **priceoye.pk** has several problems. For example, unclear customer services policy, Overwhelming competition, Scarcity of trained staff. The proposed project will overcome all of these problems by providing clear customer services. All of the services will be done under the supervision of trained staff.

* **Scope**

It will provide the customer with the opportunity to calculate Price and a search column. It will help the people to buy online things by sitting at their own homes. It will be a great advantage to the people in this critical period of Pandemic. It will also provide people with the opportunity that they can buy the genuine products as are shown on the site.

* **Functional Requirement**
* They say successful businesses begin with an idea. We say they all begin with functional and non-functional requirements for ecommerce websites.

Functional and non-functional requirements are the pillar of your project. They articulate your needs & wishes and specify the tasks for your development team. You can start writing them down soon after issuing your [RFP for ecommerce website](https://elogic.co/blog/rfp-for-ecommerce-website-hunting-out-the-best-vendor-to-replatform/) to have a clear picture of your website by the time you choose a vendor.

All of our projects at Elogic start with a brief clarifying the client’s functional and non-functional requirements. Unless a client sends a list directly to us, we issue a document for them to fill out (see it as a downloadable bonus at the end of the article!). This way, we’re sure that we understand their business needs, while the client can adjust the project budget & scope and deliver positive shopping experiences on the website.

In this article, we’ll explain the difference between the functional and non-functional requirements and present a list of the most common ones based on our clients’ real-life examples. We’ll also share some tips on how to organize the requirements and what they should look like.

Functional vs. Non-Functional: Definition & Differences

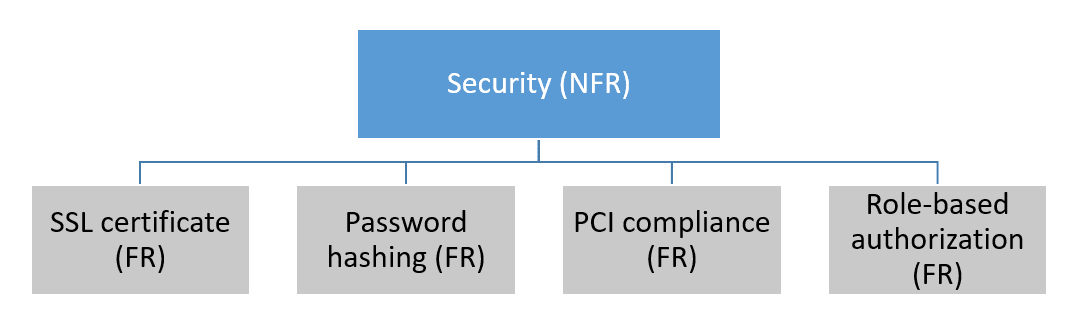
The main difference between functional and non-functional requirements lies in their scope and purpose.

**Functional requirements (FRs)** are the of your website. It is all about the functions and core operations of your e-store that enable a user to take action on the website. They can be implemented as a single website feature and form the basis of the whole software development process.

*Example*: Add the following product filtering features to our home improvement web store: price, popularity, power rate (Watt), heating area (m2), and usage (bathroom, kitchen, etc.).

**Non-functional requirements (NFRs)** are the *how* of your website. Named quality attributes of a system, they form user experience and imply some global, abstract expectations from the product. Non-functional requirements may derive from a sum of functional requirements and are implemented as a sum of web features.

*Example*: Products should be easily found and have an appealing display on the website.

Interplay of functional and non-functional requirements: many FRs make up one NFR.

*Insider tip*: Tweaking and adjusting the requirements can change the scope and budget of your project. It’s not recommended to save on functional requirements that form the core of the project, so don’t be afraid to see many of them in your website specification document. But beware of adding up the non-functional requirements which will drive up the cost of your project.

If you can afford that — great! Your store will be a customer magnet delivering optimal user experience. If, on the other hand, you’re tight on budget, consider adding only the essential non-functional requirements that will satisfy your users. Ideally, you’ll want to find the golden mean between the functional and non-functional requirements for an ecommerce website and balance them bearing in mind your business goals and objectives.

Functional and non-functional requirements are the basis of the website specification document issued to a development agency at the beginning of the project. Learn all the tips & tricks on how to write it in our related guide’s & Advice

Functional Requirements: You’re Website Core

Responsible for the system behaviour, functional requirements can be very different depending on one’s business needs and niche. For instance, fashion websites usually allow some product attributes to be selected by the customer (e.g., colour, size, etc.); travel companies may require a chat bot providing assistance to the user; luxury goods & jewellery brands need a zooming feature on a product detail page (PDP).

Here are a few must-have sets of functional requirements applicable to all ecommerce websites.

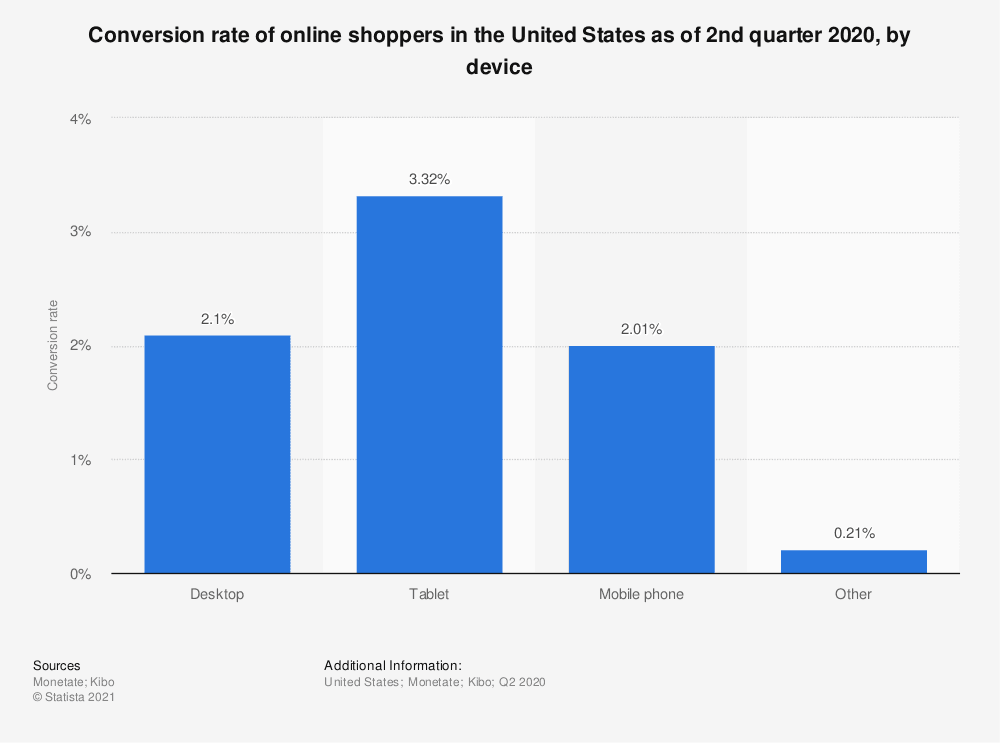
FR #1: Third-Party Integrations

Indicate which third-party software you want to add to your new website. Or maybe you’re re plat forming and wish to preserve the systems you’re using now. This requirement concerns both systems streamlining business operations (like ERP, CRM, PIM) and flexible payment gateways for your customers. Specifying a number of third-party integrations will make your [ecommerce architecture](https://elogic.co/blog/ecommerce-architecture-basic-components-all-magento-projects-need/) structured and ready for your business scaling in the future.

*FR #1 examples*:

* The website shall be integrated with ERP system and good payment solutions.
* FR #2: Mobile-Friendliness

It’s no secret that mobile responsive apps bring more traffic to the website. According to [Statista](https://www.statista.com/statistics/234884/us-online-shopper-conversion-rate-by-device/" \t "_blank), the number of customer conversions on mobile devices has also reached those on desktops in the US. So investing into a mobile-responsive feature of your website can earn you more than a few bucks, plus loyal customers shopping from the comfort of their sofas.



Study your target audience and inquire about their devices. Specify how the position of the essential buttons and options on webpages should change for better shopping experience (insider tip: place the checkout button within your customer’s thumb’s reach, for most people prefer surfing the internet with only one hand from mobile devices). Your mobile-first functional requirement should be precise so as not to confuse the developers.

*FR #2 example*: PDP should be adapted to the screens of Apple iPhone 6s and above.

FR #3: Product Attributes

Your PDP will include various product characteristics, and the development agency should know about them to implement the corresponding features. Will the customer be able to choose a product size and colour only? Do you use videos on a PDP? Will some product attributes appear in a menu (as in the [mega-menu](https://elogic.co/blog/review-of-mega-menu-magento-2-extension-for-better-website-navigation/))? If possible, write out a list of all product attributes your website should have and hand it over to your developers.

Product attributes on our client’s PDP: size, dress length, quantity, and color. Source: [Amsale](https://amsale.com/collections/amsale-bridesmaids/products/alicia" \t "_blank).

*FR #3 examples*:

* The website shall support simple and bundle product types.
* Product Images in the product detail page should have the option to zoom in.

FR #4: Order & Checkout Flow

Your functional requirements should specify how the orders are processed in your store and whether this functionality should be optimized. In particular, indicate whether you want the customer to register to make a purchase or enable a guest checkout. List the order statuses you want to have (visible both to the customer and the store admin). Explain how you want to manage B2B orders. In short, try to be as detailed about your order & checkout flow functionalities as possible.

This is also the part where you mention your discount policy and the promo codes you provide in your store, if any. Whether they are to be handled at the checkout or directly on the PDP, you should include a separate functional requirement on them.

*FR #4 examples*:

* The website shall show the following order statuses: confirmed, processing, shipped, and returned.
* After the customer is registered on the website, they should receive one extra year of warranty on the purchased order.

FR #5: Social Sharing

Online presence goes alongside social media presence in ecommerce. Allowing a user to share your website’s content on social media leads to higher brand awareness and brings you closer to your present and potential customers. Research your target audience and identify their favourite social media networks. Let your consumers share products, blog posts, and inspirational pictures by adding a corresponding button to their website.

*FR #5 examples*:

* Our product info should be shared on Facebook, Instagram, Pinterest, and LinkedIn.
* User comments from Facebook should appear on the product detail page in the “Customer Reviews” page.

**We’re following the social sharing practice ourselves at Elogic. If you’re feeling generous (and a little smarter after reading this article), why not share it on social media!**

* **Non-functional Requirements**
* **Performance Requirements**

***Internet Speed***

*It is the main requirement for our site.*

* **Safety Requirements**
* **Use a Secure Computer**

Make sure you are using a secure internet connection and not sending private information out on public computers that could contain malware. Additionally, make sure that your anti-virus, **operating system**, and web browser have the latest versions and security patches installed

* **Security Requirements**
* **VPN**

*Don’t use VPN otherwise your data will be visible to scammers*

* **User Documentation**
* *The user should have the appropriate version of windows.*
* *The system should have up to 2 GB ram minimum requirement for the application.*
* *The application should be installed on the system.*
* *So, this is the overall process of making the Online Shopping system worked, and the user can get the policy without any headache of the agent and sometimes do not need to provide a commission to them.*

***About Us***

* *This feature will provide information about the owner, team members or partners and developers, information like the ranking of the website, average daily page visits, and journey (like when the system was launched, from where does the motivation came from) will be included.*

***Contact Us***

* *In this section, we have provided the contact number and proper email address, so that the customers can anytime write to us with any queries or complaints. Also, their valuable feedbacks are always heartily welcomed. And our special team is ready to solve their problems and serve them the better way possible.*
* *Careers At*
* *This particular layout is used when we need some of the real talents to be a part of our business, by providing job openings. Using which people interested can apply and therefore will undergo further processes to get selected.*
* *Online Shopping System Project Documentation*
* *User Interface of Online Shopping System:*
* *This is one of the main tasks of the developer to design a graphical user interface that the user attracts to and can use easily; in one word it should be user-friendly. So, for this, you should have a better understanding of customers’ likes and dislikes and the features that are in trend and mesmerize the public easily, initially we need to locate the targeting people that what kind of application do they need.*
* *After getting all this information we should start to design the application.*
* **Assumptions and Dependencies**

*List any assumed factors that could affect the stated requirements. These factors are not system constraints, but areas where future changes might drive changes in the requirements. The project could be affected if these assumptions are incorrect, are not shared, or changed.*

*Also, identify any dependencies the project has on external factors. For example, if you expect to integrate into the system some components that are being developed by another project, you are dependent upon that project to supply the correctly operating components on schedule.*

* **System Architecture**

*This section should provide the complete architecture of the system with description. Diagrammatic architecture is compulsory. Also include Data Flow Diagrams in this section.*

* **Use Cases**
* **Use Case Diagrams**

****

**Use Case Description**

*Each Use Case has a description, which describes the functionality that will be built in the proposed system. The template for Use Case description is given below:*

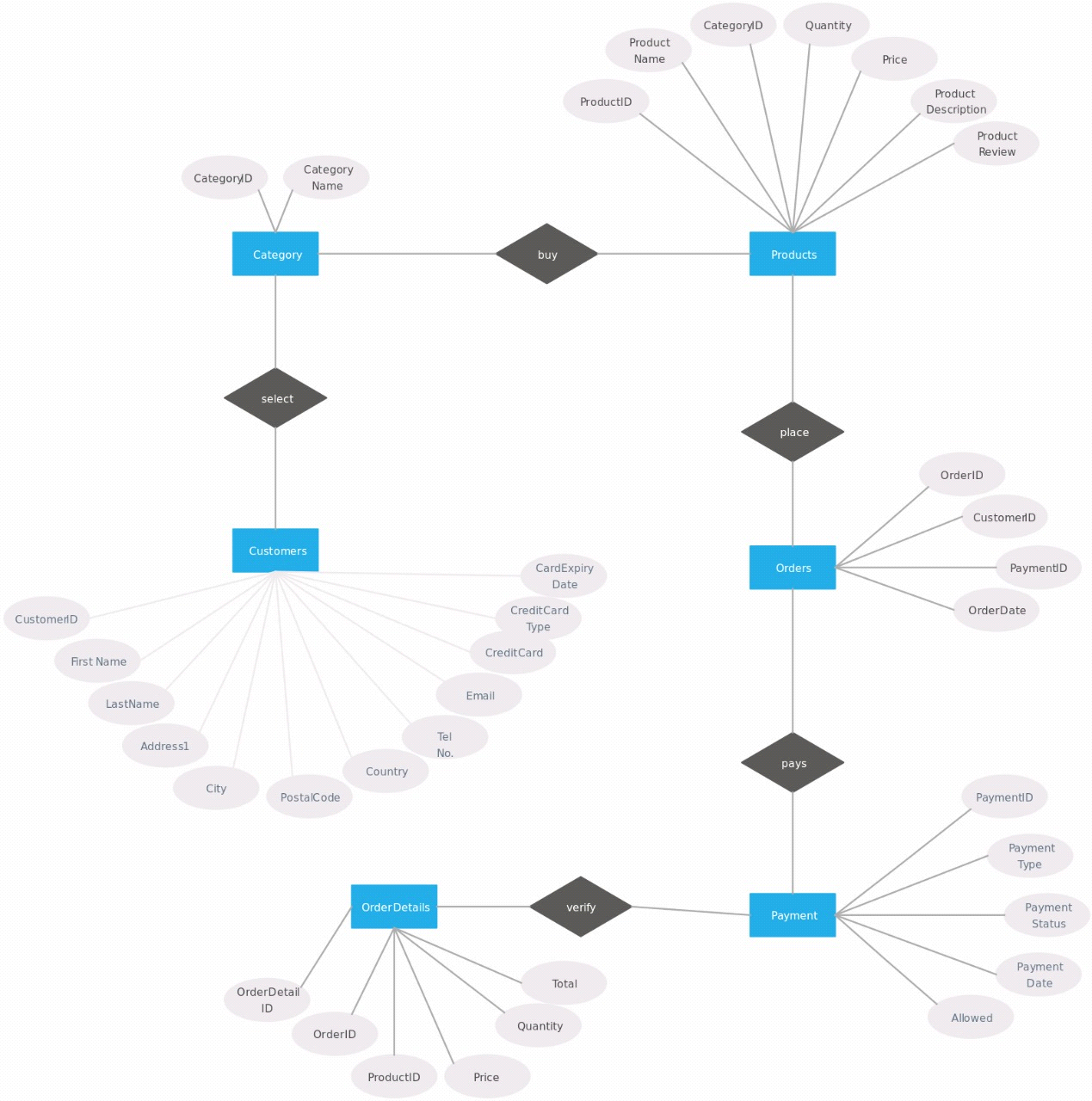
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **<Use case Id: name>** | | | | |
| **Actors:**  *<List of actors (external agents), indicating who initiated the use case>* | | | | |
| **Feature:**  *<Feature from which the use case is driven>* | | | | |
| **Use case Id:** | | *Write use case reference number.* | | |
| **Pre-condition:** | | *<List the assumptions required before this Use Case can be executed. >* | | |
| **Scenarios** | | | | |
| **Step#** | **Action** | | | **Software Reaction** |
| **1.** | *Numbered actions of the actors* | | | *Numbered description of system responses* |
| **2.** |  | | |  |
|  |  | | |  |
| **Alternate Scenarios:** *Write additional, optional, branching or iterative steps. Refer to specific action number to ensure understand ability.* | | | | |
| **1a:**  **2a:** | | | | |
| **Post Conditions** | | | | |
| **Step#** | **Description** | | | |
|  | *Sequentially list conditions expected at the completion of the use case.* | | | |
|  |  | | | |
|  |  | | | |
| **Use Case Cross referenced** | | | *<Related use cases, which use or are used by this use case>* | |
| **User Interface reference** | | | *List user interface(s) that are related to this use case. Use numbered list in case of more than one user interface elements.* | |
| **Concurrency and Response** *Give an estimate of the following*   * *Number of concurrent users* * *Expected response time of the use case* | | | | |

* **Graphical User Interfaces**

*Give a detailed account of user interfaces included in this project.*

|  |  |  |
| --- | --- | --- |
| **<User Interface Id: Title>** | | |
| **Interface Id.** | | *Write the reference number assigned to this UI.* |
| **Use case Reference** | | *Refer to the use case invoking this UI.* |
| **Snapshot** | | |
| *Include a labeled snapshot of the user interface.* | | |
| **Data dictionary reference** | | |
| **Label** | **Data dictionary identifier** | |
|  | *Refer to fields in data dictionary* | |
|  |  | |
|  |  | |
|  |  | |

* **High Level Design**
* **ER Diagram**



* **Data Dictionary**

*The convention recommended for writing the data dictionary is as follows.*

* **Data 1**

*Description (Refer to Template on next page).*

* **Data 2**

*Description (Refer to Template next page).*

***.***

***.***

***.***

***.***

* **Data n**

*Description (Refer to Template next page).*

*.*

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

|  |  |
| --- | --- |
| **< Data 1>** | |
| ***Name*** | *Give primary name of the data or control item, the data store or an external entity.* |
| ***Alias*** | *Write other names used for the first entry.* |
| ***Where-used/how-used*** | *List all processes that use the data or control item and how it is used (e.g., input to process, output from the process, as a store, as n external entity)* |
| ***Content description*** | *Notation for representing content.* |
| ***Supplementary information*** | *Other information about data types, preset values, restrictions or limitations etc.* |

*Make similar tables for all the data items.*

***The notation to develop content description is given below:***

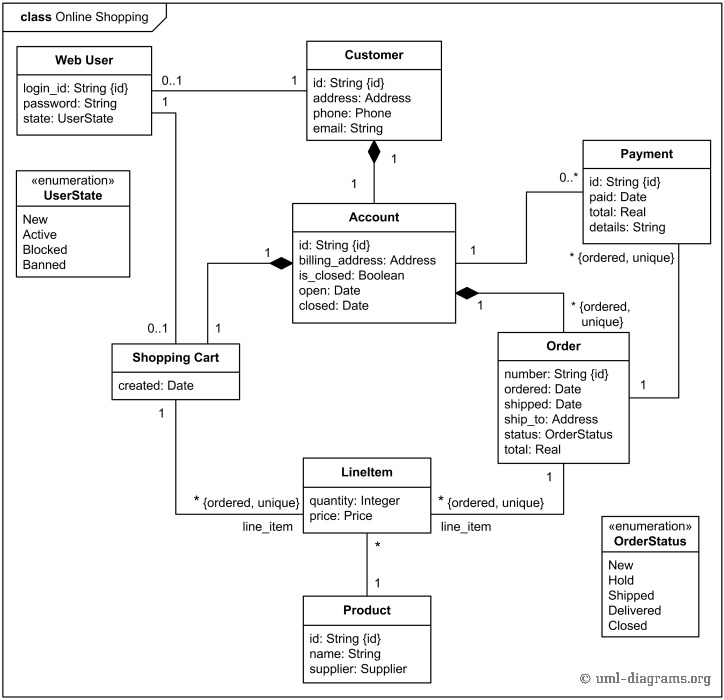
|  |  |  |
| --- | --- | --- |
| **Data construct** | **Notation** | **Meaning** |
|  |  |  |
|  | = | *is composed of* |
| *Sequence* | + | *and* |
| *Selection* | [|] | *either-or* |
| *Repetition* | {}n | *n repetitions of* |
|  | ( ) | *optional data* |
|  | \* … \* | *delimits comments* |
|  |  |  |
|  |  |  |
|  |  |  |

* **Requirements Traceability Matrix**

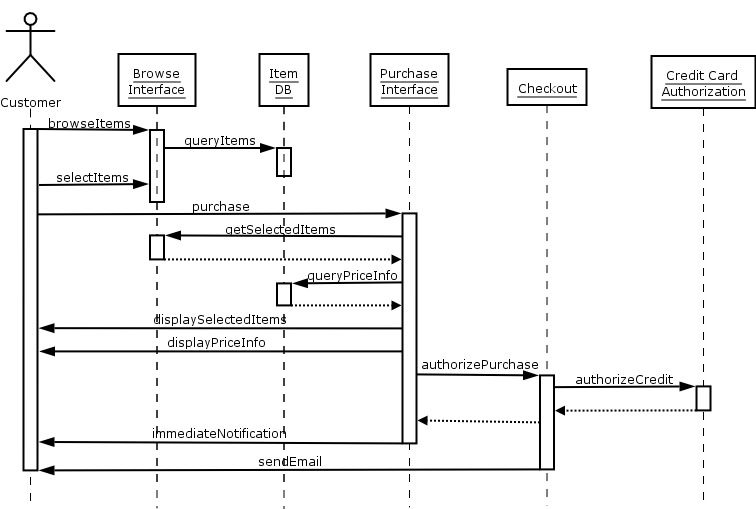
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Sr. #** | **Feature** | **Use case ID** | **UI ID** | **Priority** | **Build Number** | **Use Case Cross reference**  **(Related Use Cases)** |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |

*The columns carry the following meaning:*

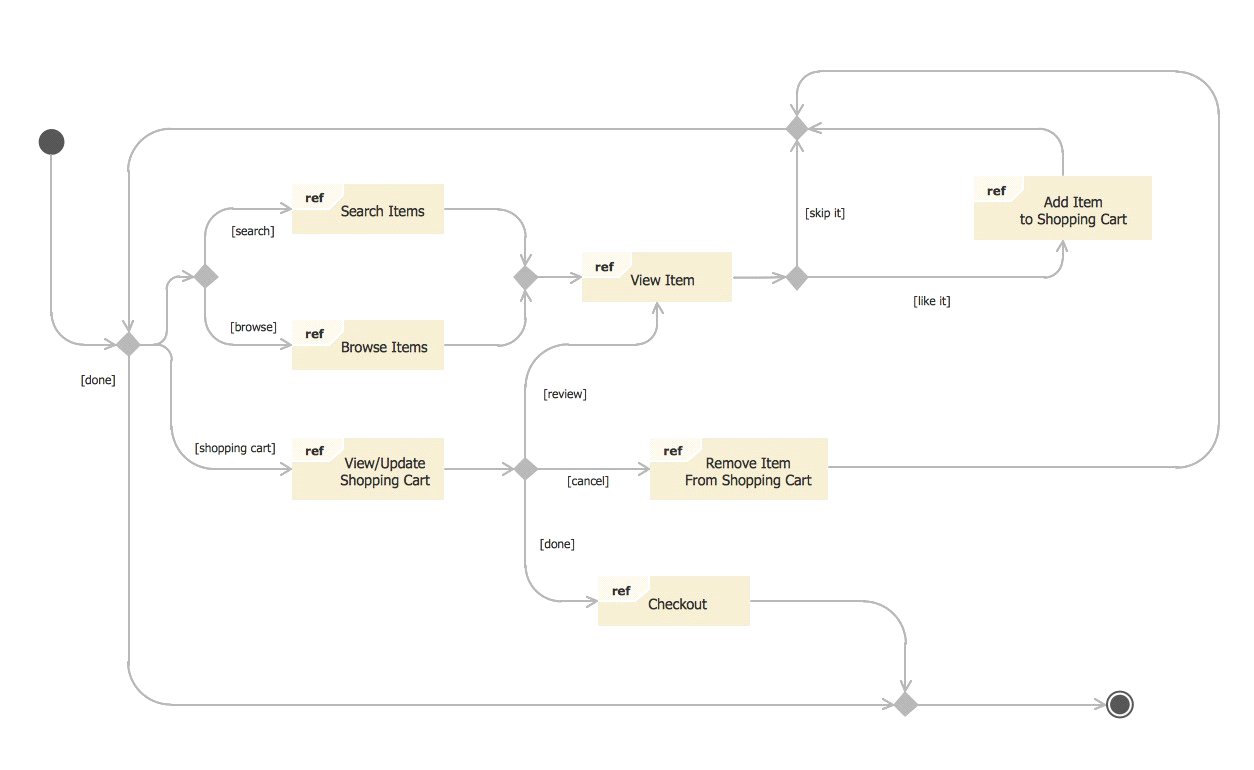
* *Feature: Lists system features based on which use cases are built.*
* *Use Case ID: Write the ID of the use case for easy lookup*
* *UI ID: Write the user interface ID for this use case.*
* *Priority: Give an appropriate rating to each use case according to its priority*
* *Build Number: Write the reference number to which this feature belongs.*
* *Use Case Cross Ref: Write the related use cases separated with commas.*
* **Class Diagram**



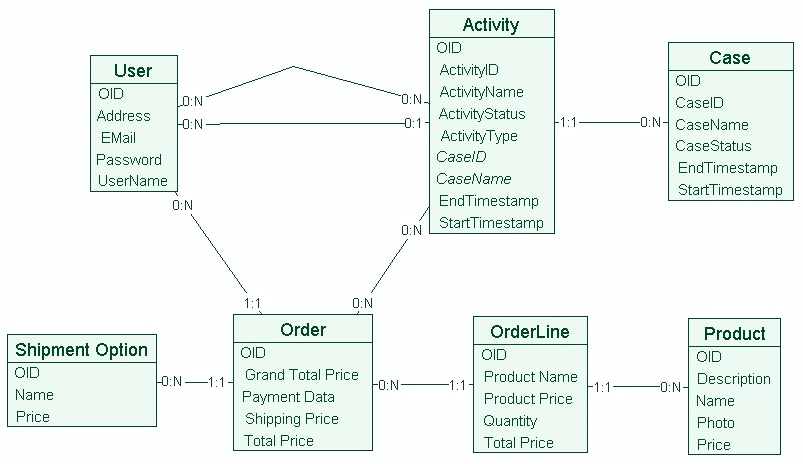
* **Sequence Diagram**



* **State Diagram**



* **Data Model**



* **Risk Analysis**

**1. Popular Places**

They say that first impressions last the longest, and the same holds true for your e-commerce website. Evaluate which pages users are landing on first and which have the highest traffic by looking at your analytics. Set time aside during regression and visual testing to ensure they’re top notch in terms of performance, functionality, usability and design.

**2. Browsing**

There are probably a few basics that every consumer expects when shopping online. Some might include login, navigation, search, filters, product pages, etc. Consider how the buyer journey would be affected if each of these places was broken. How important is filtering on a page that has 10 items? Does that change if there are 1,000? By determining how critical each element is, you can better understand what to test first.

**3. Buyer Journey**

Going from “Add to Cart” to “Order Confirmed” includes many moving parts, and if one of those parts isn’t working, it may impact your customer’s ability to complete their purchase. What’s most critical in order for the visitor to buy, and what would prevent that if broken? Testing the different ways to pay — credit card vs. PayPal — may take priority over testing auto fill for form fields, for example. Make a ranking of some of the test cases that come up during payment, and make sure the ones at the top are included in testing.

**4. Personas**

Depending on your application, you may have to consider different user states. Does someone have to log in to shop? What about to check out? Think about how the experience differs for new users compared to returning users, and account holders compared to guests. Not only do you have to test these different avenues, you have to think about the risk associated with those test cases. If an account is required, the process to create one may take on a higher priority.

**5. Privacy and Security**

Impressing your customers won’t do you much good if they don’t trust the security of your web application. Identify where personal (phone number, email address) or sensitive information (payment and password) is required. Whenever you’re storing this type of data, security testing should be of utmost importance so that leaks, hacks and breaches don’t ruin your brand's reputation.

**6. Sales and Promotions**

Do you have new pages made for a certain sale or repurposed pages you’re directing traffic to for a promotion? While the risk of these may be temporary and the tests are one-off, it’s important to keep an eye on the relevancy of your tests over time. Check often to make sure that regression tests are still relevant to meet requirements, and ensure that new tests are added as needed.

**7. Browsers and Devices**

Last but not least, you want to make sure that your website works across various browsers and devices during exploratory testing, in addition to when it undergoes a code change. Your users are visiting your site on a wide range of configurations, which means you need to test your site on them, too. By analysing which ones your visitors are on, you can focus on those instead of attempting to test them all.

There’s not enough time in the week to test every part of your web application. The good news is that planning and strategizing your testing means that you can target quality where it matters most. This way, your tests results will be more beneficial to your team and meaningful to the end user.

Alexandra McPeak is the content marketing specialist at Smart Bear, a provider of software quality tools.

* **Cost Estimation Sheet**

**(Consult** **your Project Manager for this section)**

|  |  |  |
| --- | --- | --- |
|  | **Software development cost** | 25000 to 50000 |
|  | **Packaged software** | 5000 to 8000 |
|  | **Hosting** | 4000 to 8000 |
|  | **SEO** | N/A |
|  | **Client** | N/A |
|  | **Integration** | 5000 to 10000 |
|  |  |  |
|  |  | **Total cost = 39000 to 76000** |

|  |  |  |  |
| --- | --- | --- | --- |
| **Ref. No.** | **Document Title** | **Date of Release/ Publication** | **Document Source** |
| PGBH01-2003-Proposal | Project Proposal | Oct 20, 2003 | <Give the path of your Project repository/Folder> |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

* **Appendices**

*Include supporting details that would be too distracting to include in the main body of the document.*

|  |
| --- |
| **SDS** |

**Document Information**

|  |  |
| --- | --- |
| **Category** | **Information** |
| University | UOS |
| Project | E-Dukaan |
| Document | Software Design Specification |
| Document Version | 1.0 |
| Identifier |  |
| Status | Draft |
| Author(s) | Asif Mehmood, Ali Asghar, Saad Saleem |
| Approver(s) | PM |
| Issue Date | Jan. 15, 2022 |
| Document Location |  |
| Distribution | 1. Advisor 2. PM 3. Project Office |

**Definition of Terms, Acronyms and Abbreviations**

*This section should provide the definitions of all terms, acronyms, and abbreviations required to interpret the terms used in the document properly.*

|  |  |
| --- | --- |
| **Term** | **Description** |
| ASP | Active Server Pages |
| DD | Design Specification |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

* **Introduction**
* **Purpose of Document**

We will develop the E-Dukaan Site. The objective of developing an E-Dukaan site is to provide opportunities to the public for buying online products. It will help them to have a record of all the products they want to buy in a single card. Also they can search the available products and check their price. Our site will solve the issue of traffic in the market as if most of the people buy their products online so this will decrease the traffic in the market and also solve the environmental issue.

* **Project Overview**

E-Dukaan means buying and selling of products or services over electronic systems such as the Internet and other computer networks. These online shopping sites like **Daraz.pk** and **priceoye.pk** has several problems. For example, unclear customer services policy, Overwhelming competition, Scarcity of trained staff. The proposed project will overcome all of these problems by providing clear customer services. All of the services will be done under the supervision of trained staff.

* **Scope**

It will provide the customer with the opportunity to calculate Price and a search column. It will help the people to buy online things by sitting at their own homes. It will be a great advantage to the people in this critical period of Pandemic. It will also provide people with the opportunity that they can buy the genuine products as are shown on the site.

* **Design Considerations**

*This section describes many of the issues which need to be addressed or resolved before attempting to devise a complete design solution. In other words, this section is used to formally set the groundwork for the system design.*

* **Assumptions and Dependencies**

*Assumptions and dependencies for the system and project are already captured in the FS document. This section should not repeat those issues. Instead it should deal with previously stated issues in the context of design, if appropriate, or bring up new issues that are only relevant to design.*

* **Risks and Volatile Areas**

*Discuss the most likely sources of change and risks (new requirements, technology, etc.) that would impact the design of the system. If appropriate, describe how the system will be designed to allow timely response to changes or what the contingency path is for changes.*

* **System Architecture**

*This section should provide a high-level overview of how the functionality and responsibilities of the system are partitioned and then assigned to subsystems or components. The main purpose is to gain a general understanding of how the system is decomposed, and how the individual parts work together to provide the desired functionality.*

* **System Level Architecture**

*The architecture should decompose the system at a top level in a way that provides a foundation for more detailed design work. The architecture discusses relationships and roles of system elements (subsystems, components, modules, etc.), but does not provide internal details. Areas for consideration are:*

* *System decomposition into elements*
* *The relationship between the elements*
* *Interfaces to external systems*
* *Major physical design issues such as where elements will execute*
* *Global design strategies such as error handling*

*NOTE: You may use appropriate UML diagrams (Package and Deployment diagrams) to document the overall system architecture. Similarly you can describe the top-level decomposition of the system using Component diagrams.*

* **Sub-System / Component / Module Level Architecture**

*Identify the sub-systems, component or modules of your overall software system and provide their diagrammatic view using appropriate detailed architecture diagram presenting how the system is further divided into sub systems, components and modules and the relationships and interactions between them.*

* **Sub-Component / Sub-Module Level Architecture (1…n)**

*Identify all the sub components or sub modules (if any) of the already identified modules and components. Provide their diagrammatic view using appropriate detailed architecture diagram presenting how those sub systems, modules and components are further divided into sub components and sub modules and how they interact with each other.*

* **Design Strategies**

*Describe the design strategies or decisions that impact the overall organization of the system and its high-level structures. This information should provide the reader with insights into the key abstractions and mechanisms used in the system architecture.*

* **Strategy 1…n**

*For each strategy, discuss the reasoning employed (possibly referring to previously stated design goals and principles) and any trade-offs. Areas for consideration include:*

* *Future system extension or enhancement*
* *System reuse*
* *User interface paradigms*
* *Data management (storage, distribution, persistence)*
* *Concurrency and synchronization*

COCOMO Models for E-commerce Application  
  
***Abstract-* In this paper, we are implementing different  
components of Software Cost Estimation model for E-Commerce  
Application that is built using HTML, XML and JavaScript and  
their Cost Drivers effect on the accuracy of the cost estimations  
for the software development project. Instead of it, we evaluates  
COCOMO models to find out the level of efficiency that present  
and how these can be tailored to the needs of modern software  
development projects and also present the results and findings  
after applying two of COCOMO’s sub models.  
Keywords- COCOMO I, COCOMO II, Software Cost  
Estimation, Software Cost Drivers Assessment, Analysis,  
Component Composition.  
I. INTRODUCTION**  
Software Cost Estimation models [1] can be possible by  
several estimation models such as Line of Code, Function  
Point and Constructive Cost Model (COCOMO). The original  
COCOMO model is one of the most widely practical and  
popular among the software development community because  
of some flexibility usages. Using this model we can estimates  
the cost of the software very easily. There are two types of  
COCOMO (Constructive Cost Model) i.e. COCOMO I and  
COCOMO II. The basic objective of COCOMO Model  
estimates the cost of the software and helps the developers for  
further estimation. It also helps for software decision making  
because the cost of application is the backbone of the  
application. It not only offers a cost estimation tools, but also  
provides a great amount of parameters which explain what the  
model is estimating and why it produces the estimate it does.  
COCOMO I is actually a hierarchy of three sub models and  
each sub model is progressively more details than the other.  
The first sub-model is Basic COCOMO. It is a single valued  
model and calculates the software development cost and effort  
estimation of a program by measuring Lines of Code (LOC).  
Basic COCOMO further divided into three modes based on  
the nature of the software project. First is Organic Basic  
COCOMO. It is used in small size simple software project  
that developed by small team with good experiences. Second  
is semidetached Basic COCOMO, it used in medium size  
software project developed by team with diversified levels of  
experience. Third is embedded basic COCOMO, that is used  
in massive software project with strict resource constraints  
development by multiple team acquiring the immense levels  
of experience and sophistication.  
The second sub model is intermediate COCOMO, it simply  
‘Basic COCOMO’ plus a set of subjective ‘Cost Drivers’.  
These drivers are used to access product, computer and project  
attributes of a software project. The evaluator uses a six level  
scale to decide where each attribute fall. When an attribute is  
accessed, it produces an adjustment factor. This adjustment  
factors are multiplied together and give an Effort Adjustment  
Factor (EAF) that is usually equal to a value between 0.9 and  
1.4. The EAF is then mathematically applied on all basic  
COCOMO’s formulas.  
The third sub model is detailed COCOMO, as the name  
indicates, it produced the most accurate cost estimation of all  
three sub models of COCOMO I. It combines Basic and  
Intermediate COCOMO together boosted by an assessment of  
every cost Driver’s impact on each stage of ‘Barry Boehm’s  
software engineering processes.  
On the other hand COCOMO II divided into four sub models.  
Each sub model is based on different input and estimates the  
effort of different activities of a software project. Application  
composition is the first sub-model. It estimates the effort of  
prototype systems developed using script, database  
programming etc. Second sub-model is ‘Early Design’ that  
calculates the initial effort based on system requirement and  
design option, and uses function points as input. Third sub model is “Reuse’, it estimates the effort of integrating reusable  
automatically generated line of code as an input. Fourth sub model is ‘Post Architectural. It estimates the development   
effort of system design specifications and use line of source  
code as an input.  
**II. RELATED WORKS**  
There are various evaluation criteria given by various  
personalities but the evaluation criteria proposed by Boehm  
for the validity of process models gives effective results. The  
articles explain the strength and weakness of various  
cost estimation technique for the period of 1965 to 2005.  
COCOMO II was an excellent model up to 2005 but it did not  
enfold the new requirement and development styles for the  
rudeness or estimation of cost, because these days the  
technology are going to advance and we are using advance  
technology and developed the application and during  
development process the estimation of the application are very  
difficult using COCOMO model. The authors discuss different  
software cost estimation technique and highlighted various hot  
area and challenges of the research in the field of software  
cost estimation.  
In article [3], it is emphasized that there should be a need to  
research more in this field to open new horizons for research.  
Reusability of components in component based development   
(CBD) is illustrated in article. This discusses and  
compared different architectures of CBD and mentioned the  
advantages and disadvantages of CBD. A comparison, of  
component based development (CBD) with other traditional  
software development practices, is also provided.  
Sulci and Baruchelli highlighted the importance of  
standardization of components for the software reusability.  
The discussion of this research paper was to find how total  
development cost of a software system affected on the basis of  
component-based software engineering. The two main factors  
that affect the standardization cost of a component have been  
explained. According to them, the cost of the standardization  
of component(s) must be included during the cost-benefit  
analysis of a software system.  
Highlighted the pertinent issues of software  
reusability and its high level guidelines for component based  
development on the basis of CBSE. He mentioned that how  
much reusability resulted to improve product reliability and to  
reduce overall software development cost.  
Wrote a report for the validation of the component based method (CBM) for software size estimation by the  
analysis of 46 projects. Then the complete process of this  
analysis and different techniques of analysis was mentioned.  
Relationship of LOC (Line of Code) and NOC (No. of  
Component) was carried out with suitable examples.  
Comparison of CBM and a Global Method (Mark II) was also  
provided.  
**III. RESEARCH PROBLEMS**based on the literature survey and study, a number of  
discussions are reported for the effectiveness and accuracy of  
COCOMO models. This paper is written to find out the  
accuracy of cost estimation of both models when applied on a  
specific project. Further we try to find out, what is the impact  
of cost drivers during the system development life cycle  
phases.  
Validate and the accuracy of the cost estimations [11] of  
COCOMO models for projects is important that are built  
using HTML, XML and JavaScript. Hence, we will not only  
find out how accurate and reliable they are, but also whether  
they are suitable for estimating HTML, XML and JavaScript  
Code. **IV. EVALUATION OF COST ESTIMATION**this paper follows the principles and creating an E-Commerce  
web application of an online bookstore. The application  
consists of fourteen web pages written in HTML, XML and  
JavaScript. All fourteen pages were fully designed to have  
different content and perform different web tasks. Then, they  
were coded and connected together according to their design.  
The pages are a demo experience of how a real user would  
buy a book online.  
The sub section 4.1 covers the COCOMO I whereas  
COCOMO II is covered in the sub section 4.2 subsequently.  
V. APPLYING COCOMO I  
Sub-model Used: Basic COCOMO I  
Model Used: Organic  
Formula Used:  
(1)

|  |  |
| --- | --- |
| (2) | Calculating Total LOC: |

|  |  |
| --- | --- |
| Webpage Name | Numbers of Lines of Code |
| About Us | 101 LOC |
| Book Details | 119 LOC |
| Categories | 376 LOC |
| Home | 91 LOC |
| Contact Us | 114 LOC |
| Feed Back | 267 LOC |
| Index | 276 LOC |
| My Account | 114 LOC |
| Payment | 245 LOC |
| Search Results | 327 LOC |
| Shopping Cart | 157 LOC |

|  |  |
| --- | --- |
| Sign In | 118 LOC |
| Sign Up | 190 LOC |
| Verification | 146 LOC |
| Total=2918 LOC, 2.918 KLOC |  |

TABLE I: HTML PAGES, 14 PAGES IN TOTAL  
Estimating Effort:  
Effort = 9.851 MM  
Estimating Time:  
Time = 2.5  
Time = 5.963 Months  
Sub-model Used: Intermediate COCOMO I  
Mode Used: Organic  
Formula Used:  
Time = 2.5  
Cost Drivers:  
• Product Attributes:  
1. RELY – Required Software Reliability.  
2. DATA – Data Base Size.  
3. CPLX- Product Complexity.  
• Computer Attributes:  
4. TIME- Execution Time.  
5. STOR- Main Storage.  
6. VIRT- Virtual Machine Volatility.  
7. TURN- Computer Turn Around Time.  
• Personal Attributes  
8 ACAP-Analyst Capability.  
9 AEXP-Application Experience.  
10 PCAP- Programmer Capability.  
11 VEXP- Virtual Machine Experience.  
12 LEXP- Programming Language Experience.  
• Project Attributes  
13 MODP- Use of Modern Programming Practices.  
14 TOOL- Use of Software Tool.  
15 SCED- Required Development Schedule.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Very Low | Low | Normal | High | Very High |
| RELY | 1.15 |  |  |  |
| DATA | 1.16 |  |  |  |
| CPLX | 1.30 |  |  |  |
| TIME | 0.85 |  |  |  |
| STOR | 1.21 |  |  |  |
| VIRT | 1.30 |  |  |  |
| TURN | 1.15 |  |  |  |
| ACAP | 0.86 |  |  |  |
| AEXP | 0.80 |  |  |  |
| DCAP | 1.0 |  |  |  |
| VEXP | 0.90 |  |  |  |
| LEXP | 0.95 |  |  |  |
| MODP | 1.0 |  |  |  |
| TOOL | 0.83 |  |  |  |
| SCED | 0.85 |  |  |  |

**CONCLUSION**

'Application Composition' is intended for prototype projects,

Where the project is built by composing components called

'Object Points'. It not only consider the cost drivers but also

Project environment's characteristics like developer's

Experience and capability, CASE tool's maturity and

Capability, number of screens, and number of system

Generated reports. So, each component (object) is customized

And then attached to whole body of the project in a different

Way and with a different level of challenge. As mentioned in

Previous section 4, 90% of the project is reused components

From existing e-commerce applications. The relation between

'Reuse' and 'Effort' is disproportional i.e. the more components

Are reused (90% out of 100%), the less effort (1 Man Month)

Is required.

In the other hand, 'Early Design' model is similar to 'Organic

Intermediate COCOMO'. In each model, more information is

Uncovered as initial stages of the project are concluded and

Design stages are initiated. However, only a rough system

Design is required to make early estimations. While comparing

The value of effort estimated by 'Application Composition' and

The value of effort estimated by 'Early Design', more 15.118

Man effort is needed to meet the increasing size of the project.

When 'Reuse' model is used HTML code can be generated

Using various code generators. It may appear that 'Early

Design' model and 'Reuse' model are similar, because they

Both estimate reusable components, but that is not the case. In

Fact, the estimator using 'Early Design' needs neither to

Understand the reusable components nor to modify it. The

Estimator simply just uses them. Each time the generated code

Is studied and then refined, the cost of the component will

Decrease.

'Post-Architectural' model is used once an initial architectural

Design of the system is available. The model uses the same

Formula as 'Reuse' model uses. However, the estimation is the

Most accurate and realistic among others, because ten more

Cost drivers are uncovered and used in the formula. In section

4.2, 16.118 man efforts are needed to develop the e-commerce

Application.

* **References**

[**https://www.quicksprout.com/create-ecommerce-website/**](https://www.quicksprout.com/create-ecommerce-website/)

[**https://www.shopify.com/tour/ecommerce-website**](https://www.shopify.com/tour/ecommerce-website)

[**https://www.wpbeginner.com/wp-tutorials/how-to-start-an-online-store/**](https://www.wpbeginner.com/wp-tutorials/how-to-start-an-online-store/)

|  |  |  |  |
| --- | --- | --- | --- |
| **Ref. No.** | **Document Title** | **Date of Release/ Publication** | **Document Source** |
| 1 | Prof. Betty H.C. Cheng, “Homework 2”, CSE 435, East Lansing, MI, | 09/2007 | https://www.cse.msu.edu/~chengb/RE-491/Papers/SRS-BECS-2007.pdf |
| 2 | Prof. Betty H.C. Cheng, “Requirements Assignment”, CSE 435, East Lansing, MI, | 2007 | https://www.cse.msu.edu/~chengb/RE-491/Papers/SRS-BECS-2007.pdf |
| 3 | Mr. Borzoo Bonakdarpour, Elicitation Meeting, September 25th 2007. | 09/2007 | https://www.cse.msu.edu/~chengb/RE-491/Papers/SRS-BECS-2007.pdf |
| 4 | IEEE-SA Standards Board, “IEEE Recommended Practice for Software Requirements Specifications”, Software Engineering Standards Committee of the IEEE Computer Society | June 25th 1998 | https://www.cse.msu.edu/~chengb/RE-491/Papers/SRS-BECS-2007.pdf |

[1] Li MS, He M, Yang D, et al., *“Software cost estimation method  
and application,”* Journal of Software, vol. 18, no. 4, pp. 775-795,  
2007.  
[2] Boehm, B. W. and R. Valerdi. *Achievements and Challenges in  
Cocomo-Based Software Resource Estimation* published by IEEE  
Computer Society. 74-83 (2008).  
[3] Boehm, B. W. *An Overview of the COCOMO 2.0 Software Cost  
Model* (1999).  
[4] Zaid, A., M. H. Selamat, A. A. A. Ghani, R. Atan and K. T. Wei.  
*Issues in Software Cost Estimation, IJCSNS Int J of Computer  
Science and Network Security*, 8(11): 350-356 (2008).  
[5] Succi, G. and F. Baruchelli. *The Cost of Standardizing  
Components for Software Reuse*, Standard View 5(2) (1997).  
[6] Gill, N. S*. Reusability Issues in Component-Based Development*,  
ACM SIGSOFT Software Engineering Notes, 28(4): 4 – 4 (2003).  
[7] Clemente, P. J. and J. Hernández. *Aspect Component Based  
Software Engineering,* University Extremadura. 1-4 Spain (2001).  
[8] Frakes, W. B. and K. Kang*. Software Reuse Research: Status and  
Future*, IEEE Transactions on Software Engineering, 31(7): 529-  
536 (2005).  
[9] Dolado, J. J. *A Validation of the Component-Based Method for  
Software Size Estimation*, IEEE Transactions on Software  
Engineering, 26(10): 1006-1021 (2000).  
[10] Zaid, A., M. H. Selamat, A. A. A. Ghani, R. Atan and K. T. Wei.  
*Issues in Software Cost Estimation*, IJCSNS International Journal  
of Computer Science and Network Security, 8(11): 350-356  
(2008).  
[11] Yongchang Ren. Et al, “*B to C E-commerce Platform Software  
Development Cost and Schedule Estimation Method*” by IEEE  
(2010).