**REPUBLIC OF CAMEROON**

***Peace - Work - Fatherland***

****

**Personal Expense Tracker with Data Analysis**

Tel : (237) 620 25 60 69/ 654-486-985

E-mail : realize.contactus@gmail.com

Web Site : www.realize.com

**REPUBLIC OF CAMEROON**

***Peace-Work-Fatherland***

**AFRICAN INSTITUTE OF COMPUTER SCIENCE-CAMEROON**

**P.O. Box: 13719**

Tel : (+237) 22 729957/22 72 99 58

E-mail : [contact@iaicameroun.com](mailto:contact@iaicameroun.com)

Web site: www.iaicameroun .com



**INTERNSHIP REPORT**

**Conception and Realization of a Personal Budget Management Application**

Internship carried out from the 01st July to 30TH September 2024

In view of obtaining a **Higher Technician Diploma** **(HTD) in** Computer Science.

Option: **Software Engineering**

Written By:

KANJOUH ESTHER NKWENTI

Level II student at AICS Cameroon

**Supervised By:**

**Professional Supervisor**

Mr. Nguh Prince

Computer Engineer and Lecturer at AICS

**Co**

**Academic Supervisor**

Mr. Messio

Computer Engineer and Lecturer at AICS

Academic Year 2023 - 2024

# DEDICATION

**TO MY FAMILY**

# ACKNOWLEDGEMENTS

As a prelude to this internship report we wish to express our sincere thanks to those who have assisted us and have contributed to the accomplishment of this report as well as the success of this great academic year. We would like specially to thank:

* We give thanks to the almighty God who made it possible without whom nothing would have been done.
* The Resident Representative of the African Institute of Computer Sciences, **Mr. Armand Claude Abanda.**
* **Mr. Messio,** my academic supervisor and lecturer at AICS-Cameroon for her availability, devices and moral support through the internship period which led to the realization of this project.
* **Mr. Nguh Prince** our professional supervisor for his availability, help, advices and moral support throughout the internship period which led to the realization of this project.
* To all our teachers of the African Institute of Computer Sciences for their support.
* The amazing family at Realize for their endless love and support during this internship period.
* To my family and my parents for their guidance and support without which we could not attain the end of this work.
* To all our classmates and friends at AICS, for their knowledge, ideas, collaboration and most importantly the unforgettable moments spent together.

# SUMMARY

[DEDICATION ii](#_Toc173582000)

[ACKNOWLEDGEMENTS iii](#_Toc173582001)

[SUMMARY iv](#_Toc173582002)

[LIST OF FIGURES v](#_Toc173582003)

[LIST OF TABLES vii](#_Toc173582004)

[LIST OF ABBREVATIONS viii](#_Toc173582005)

[ABSTRACT ix](#_Toc173582006)

[RESUME x](#_Toc173582007)

[GENERAL INTRODUCTION 1](#_Toc173582008)

[PART ONE: INSERTION PHASE 3](#_Toc173582009)

[PART TWO: EXISTING SYSTEM 15](#_Toc173582010)

[INTRODUCTION 16](#_Toc173582011)

[PART 3: SPECIFICATION BOOK 23](#_Toc173582012)

[PART THREE: ANALYSIS PHASE 38](#_Toc173582013)

[PART FOUR: CONCEPTION PHASE 80](#_Toc173582014)

[PART FIVE: REALIZATION PHASE 97](#_Toc173582015)

[PART SIX: USER’S GUIDE 105](#_Toc173582016)

[GENERAL CONCLUSION 114](#_Toc173582017)

[PART SEVEN:ANNEX: 115](#_Toc173582018)

[BIBLIOGRAPHY 118](#_Toc173582019)

[WEBOGRAPHY 119](#_Toc173582020)

# LIST OF FIGURES

[Figure 1: Geographical location 7](#_Toc176962458)

[Figure 2:Functional branch of realize 11](#_Toc176962459)

[Figure 3:Track Unified Process (Source:Google) 44](#_Toc176962460)

[Figure 4:UML Diagram (Source Google) 48](#_Toc176962461)

[Figure 5:formalism of a use case diagram 52](#_Toc176962462)

[Figure 6:use case diagram 55](#_Toc176962463)

[Figure 7:specific use case diagram for manage records 56](#_Toc176962464)

[Figure 8:specific use case diagram for manage cash reserves 57](#_Toc176962465)

[Figure 9:formalism of an activity diagram 60](#_Toc176962466)

[Figure 10:activity diagram for authentication 64](#_Toc176962467)

[Figure 11:activity diagram for set budget 65](#_Toc176962468)

[Figure 12:activity diagram for add record 66](#_Toc176962469)

[Figure 13:activity diagram for record deletion 67](#_Toc176962470)

[Figure 14:formalism of a state machine diagram 68](#_Toc176962471)

[Figure 15:state machine diagram for record 70](#_Toc176962472)

[Figure 16:sequence diagram formalism 71](#_Toc176962473)

[Figure 17:sequence diagram for authentication 73](#_Toc176962474)

[Figure 18:sequence diagram for add records 74](#_Toc176962475)

[Figure 19:sequence diagram for edit records 75](#_Toc176962476)

[Figure 20:formalism of a communication diagram 76](#_Toc176962477)

[Figure 21:communication diagram for add records 78](#_Toc176962478)

[Figure 22:communication diagram for edit records 78](#_Toc176962479)

[Figure 23:Three-tier architecture (source: Softwaretestingclass.com) 82](#_Toc176962480)

[Figure 24:formalism of a package diagram 85](#_Toc176962481)

[Figure 25:Formalism of a deployment diagram 87](#_Toc176962482)

[Figure 26:formalism of a component diagram 89](#_Toc176962483)

[Figure 27:formalism of a class diagram 90](#_Toc176962484)

[Figure 28:components of a class diagram 91](#_Toc176962485)

[Figure 29:: Formalism of an object diagram 93](#_Toc176962486)

[Figure 30:object diagram 96](#_Toc176962487)

[Figure 31:Model view controller (source: https://developer.mozilla.org/en-US/docs/Glossary/MVC ) 103](#_Toc176962488)

# LIST OF TABLES

[Table 1:Hardware resources of realize 12](#_Toc176963676)

[Table 2: software resources of realize(SOURCE: MERCURIAL 2022) 13](#_Toc176963677)

[Table 3:crticism of the existing system 19](#_Toc176963678)

[Table 4:human resource (MERCURIAL 2023) 31](#_Toc176963679)

[Table 5:software resource (MERCURIAL 2023) 32](#_Toc176963680)

[Table 6: Hardware resources (MERCURIAL 2023) 33](#_Toc176963681)

[Table 7:total project cost 34](#_Toc176963682)

[Table 8: Project team 35](#_Toc176963683)

[Table 9: Differences Between MERISE and UML 41](#_Toc176963684)

[Table 10:differences between 2tup and up 42](#_Toc176963685)

[Table 11: Risk analysis of the project 50](#_Toc176963686)

[Table 12: Components of a use case diagram 53](#_Toc176963687)

[Table 13: Textual description of use case create Expense 58](#_Toc176963688)

[Table 14:textual description of authentication 59](#_Toc176963689)

[Table 15:components of an activity diagram 61](#_Toc176963690)

[Table 16: Components of state machine Diagram 69](#_Toc176963691)

[Table 17: Components of sequence Diagram 72](#_Toc176963692)

[Table 18: Components of communication diagram 77](#_Toc176963693)

[Table 19: Components of a package diagram 86](#_Toc176963694)

[Table 20: Components of Deployment diagram 88](#_Toc176963695)

[Table 21: Components of object diagram 94](#_Toc176963696)

[Table 22: Software resources 100](#_Toc176963697)

# LIST OF ABBREVATIONS

* 2TUP: Two-Track Unified Processes.
* AICS: African Institute of Computer Sciences
* DB: Database.
* DBMS: Database Management System.
* CSS: Cascading Style Sheet.
* IAI: Institut Africaine de l’informatique
* UML Unified Modelling language
* Merise: Methode d’Etude et de Realisation Informatique pour les Systemes d’Entreprises.

# ABSTRACT

During a three months internship, we diligently worked on the theme titled “Analysis, Design, and Implementation of a Personal Budget management Mobile Application”. This report presents the development of a personal expense tracking application designed to assist users in effectively managing their finances. This theme is a specific solution that arises from an observation within our society, where some individuals struggle with managing their money, particularly their budget.

The application facilitates the input, categorization, and storage of financial transactions, providing a comprehensive overview of spending habits. By employing data analysis techniques, the system generates insightful reports on expenditure patterns, identifying areas of potential savings and opportunities for financial growth. The application aims to empower users to make informed financial decisions, achieve financial goals, and ultimately enhance their overall financial well-being.

For its implementation, we used UML for the analysis and modelling, and JavaScript (React and Node) for creating the application. Additionally, we adopted the 2TUP process for its meticulous approach, allowing us to progress step by step while ensuring advancement through various tests.

Keywords

* Budget
* Expense
* Income

# RESUME

Lors d’un stage de trois mois, nous avons travaillé avec diligence sur le thème intitulé « Analyse, conception et mise en œuvre d’une application mobile de gestion de budget personnel ». Ce rapport présente le développement d’une application de suivi des dépenses personnelles conçue pour aider les utilisateurs à gérer efficacement leurs finances. Ce thème est une solution spécifique qui découle d’un constat au sein de notre société, où certains individus ont du mal à gérer leur argent, en particulier leur budget.

L’application facilite la saisie, la catégorisation et le stockage des transactions financières, offrant un aperçu complet des habitudes de dépenses. En utilisant des techniques d’analyse de données, le système génère des rapports perspicaces sur les habitudes de dépenses, identifiant les domaines d’économies potentielles et les opportunités de croissance financière. L’application vise à permettre aux utilisateurs de prendre des décisions financières éclairées, d’atteindre des objectifs financiers et, en fin de compte, d’améliorer leur bien-être financier global.

Pour sa mise en œuvre, nous avons utilisé UML pour l’analyse et la modélisation, et JavaScript (React et Node) pour la création de l’application. De plus, nous avons adopté le processus 2TUP pour son approche méticuleuse, nous permettant de progresser étape par étape tout en assurant l’avancement à travers divers tests.

Mots clés

* Budget
* Dépenses
* Revenus

.

# GENERAL INTRODUCTION

Today our generation uses computers in order to do more tasks. It is for this reason that there is a need of professionals in the domain of computer sciences. Thanks to AICS (PAUL BIYA Technological Center of Excellence), we were able to undergo this training. During the second year of training, our institute recommends an internship for students in an enterprise in order to master theoretical and practical knowledge and to familiarize themselves with the professional milieu. It is for this reason that we requested an internship in an enterprise, REALIZE who opened its doors to us for a period of 3 months. This enterprise develops mobile, desktop and web applications, as well as other computer related services.

The work presented in this internship aims to design and implement a system that brings together the various functionalities required for an Expense Tracker Platform with Analysis and confronts the drawbacks of existing solutions. The achievement of this theme was carried out under six main sections which are:

**Insertion phase:** Here, we present the internship site and the theme which we will be working on.

**Existing system:** Here, we describe the structure of the existing system.

**Specification Book:** At this level, we identify the needs of the customer considering the timing and cost of the project.

**Analysis phase:** Will present the analysis of our project using UML as a modelling language, and as process, we used 2TUP.

**Conception phase:** Will present the generic conception, preliminary conception and detailed conception.

**Realization Phase:** Presents the different activities carried out to create the application.

**User Guide:** Presents the conditions necessary for the use of the application and how it is to be used.

# PART ONE: INSERTION PHASE

Preamble

This phase presents the details of how we were integrated in the host company, the company presentation and organization.

Overview

INTRODUCTION

1. WELCOME AND INTEGRATION
2. GENERAL PRESENTATION OF THE COMPANY
3. ORGANISATION OF THE XOMPANY
4. HARDWARE AND SOFTWARE RESOURCES OF THE COMPANY
5. BRIEF PRESENTATION OF THE PROJECT THEME

CONCLUSION

**INTRODUCTION**

The insertion phase is a period (generally of 02 weeks) reserve for the different interns to discover and to familiarize with the working environment. Here, we got to know about the staffs, the different hardware and software resources used, the different departments which constitute the enterprise, how the company function both internally and externally and we were introduced to our work space. During this period, we were also attributed an internship master often called professional supervisor and a theme. We also had a time to discuss amongst us interns on topics like what we love doing most, what we dislike, our believes and experiences. We shared about different realizations and failures in life.

## **WELCOME AND INTEGRATRION**

### Welcome

We arrived at REALIZE on Monday July 2024 at 8:00am. We were welcomed, by **Mr Ndabose Daniel** the enterprise frontend developer, who introduced us to our work space and his collaborators. Later run, he gave us the task to do some findings on the possible internship themes that we can work on.

### Integration

A working day at REALIZE start from 08:00am to 04:30pm. Our tasks in the company generally follow a predefine routine. Every day we were to write five pages of our internship report, to complete the assigned task for the day. It was also an opportunity to present our difficulties and challenges to everyone and receive directives.

## GENERAL PRESENTATION OF THE COMPANY

### Geographical location



Figure 1: Geographical location

### History

Realize is a Cameroon based non-governmental tech start-up institute founded by **NDELOGAKEH DANIEL** in 2022 which proposes IT solutions and empowers the use on new technologies in Cameroon.

### Mission

The mission of Realize mainly relates to the empowerment and the perpetuation of Technology use. These missions include:

* Design and realize mobile-oriented software for companies and individuals;
* Assist newborn companies for quick growth using computer sciences;
* Provide training and certifications to improve qualified human resources in many fields of study;
* Take an active part in the sustainable development of the world through innovative solutions and virtual reality.

### Vision

At Realize, we believe that the true potential of every organization and individual can be unlocked through the harmonious fusion of analysis, development, realization, and data analytics. Our vision propels us forward, guiding us to make a lasting impact on the digital landscape and shape a future where innovation knows no bounds.

### Activities

The activities of Realize range from computer sciences, engineering, and training. We can outline the following:

* Conception realization, and hosting of websites;
* Software development and maintenance;
* Training in Software related fields;
* Conception and realization of multimedia;
* IT consulting and innovation
* IT support

## ORGANISATION OF THE ENTERPRISE

### Administrative Organization of REALIZE

REALIZE is administratively organized as follows;

* 1. **Administrative organization**

Realize is administratively organized as follows:

1. **The General management**

This is the highest level of the company, which takes care of the following:

* + Proper functioning of each department.
  + Define project strategies.
  + Provide leadership and guidance to the company’s employees.
  + Makes critical decision that could affect the company’s operations or reputation
  + Severe as a point of contact for key stakeholders such as investors etc.

1. **Human resource department**

This department is in charge of the following:

* + Recruitment and hiring.
  + Acts as a liaison between employees and management;
  + Manages employee compensation and benefits programs;
  + Develops and implements company policies and procedures, ensuring compliance with employment laws and regulations.
  + Manages administrative tasks related to employee records, data management, and HR systems.

1. **Communication Department**

This department is in charge of the following:

* + Handles public relations activities, which involve managing the company’s reputation and image.
  + Responsible for crafting and delivering messages to external stakeholders such as customers etc.
  + It fosters effective communication within the company.
  + They create and manage content across different platforms and channels.

Prepare and present accurate and timely financial reports to management, stakeholders, and regulatory authorities.

* + Responsible for managing costs and expenses within the organization.
  + Ensures compliance with tax laws and regulations. Etc.

1. **Technical department**

This department is in charge of the following:

* + Managing the company’s technological infrastructure, including networks and hardware.
  + Play a role in managing and maintaining the company’s data.
  + Implementing and maintaining cybersecurity measures to protect the company’s digital assets from potential threats.
  + Interacts with technology vendors and manages relationships with external service providers. Etc.

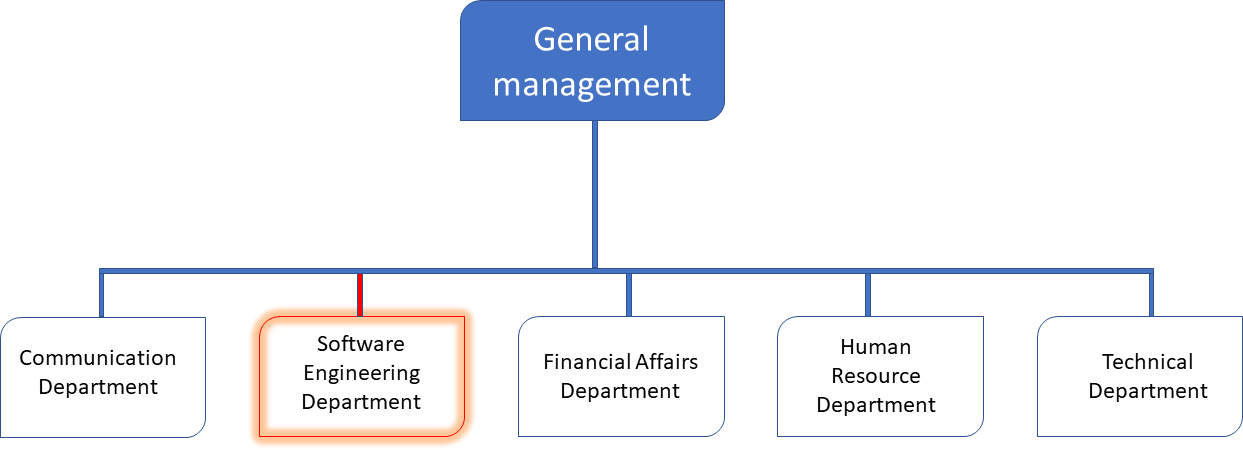
1. **Software Engineering Department**

This department is in charge of the following:

* + Primarily responsible for developing software applications and systems.
  + Responsible for maintaining and supporting software applications throughout their lifecycle.
  + Evaluation and realization of projects etc.
  1. **FUNCTIONAL ORGANIZATION OF REALIZE**

The functional branch of realize is organized as follows

Figure 2:Functional branch of realize



HARDWARE AND SOFTWARE RESOURCES OF THE COMPANY

### Hardware Resources

Table 1:Hardware resources of realize

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Designation | Quantity | Characteristics | Observation |
| 1 | MacBook Pro | 1 | APPLE | Good |
| 2 | Modem | 1 | CAMTEL | Good |
| 3 | LCD Screen | 1 | HP | Excellent |
| 4 | Training equipment and office furniture | / | / | / |

### Software Resources

As an enterprise specialized in IT, REALIZE also comprises of software resources such as;

Table 2: software resources of realize(SOURCE: MERCURIAL 2022)

|  |  |
| --- | --- |
| Designation | Software |
| Operation system | Windows 10, Mac OS, parrot Linux |
| Design tools | Photoshop |
| Integrated development environment (IDE) | Visual studio code, |
| Text editor | Sublime text, Notepad++ |
| Database management system (DBMS) | MongoDB, PostgreSQL, MySQL |
| Web browser | Google chrome, Microsoft edge |
| Document editor | Microsoft office word |
| Presentation | Microsoft office PowerPoint |

## BRIEF PRESENTATION OF THE PROJECT THEME

During our insertion phase in REALIZE, we were asked to look for themes which solves a specific problem with innovative solutions. Our professional supervisor took some time to look into what we proposed as themes and he also proposed some themes ideas. Finally, we came up with the proposed theme “CONCEPTION AND REALIZATION OF A PERSONAL BUDGET MANAGEMENT APPLICATION”. This Application will help the users to be more financial discipline, improved financial forecasting, identify spending habits, setting budgets and managing debts.

# PART TWO: EXISTING SYSTEM

## INTRODUCTION

When undertaking the analysis and management of a project, it is essential to understand the current state, also called the existing system. Studying the existing system allows us to evaluate the current situation, identifying strengths and weaknesses as well as determining opportunities for improvement. This is a crucial step that allows us to lay the foundations for the development and implementation of a successful project.

## THEME PRESENTATION

In the context where personal financial management is a major issue, the need for a mobile application for personal budget management is necessary. It is with this in mind that we took the analysis, design and creation of an application that aims to offer users a practical and intuitive tool to effectively manage their finances.

The main objective of this application is to help the end users track their income, expenses and overall budget in a simple and accessible way with a friendly user interface and advanced features.

The analysis of this project began with an in-depth study of the user’s needs and expectations regarding personal budget management. We identified key functionalities such as expense tracking, income tracking, transaction category, creation of personalized budgets, generation of reports and visualization of financial data. Based on these, we built advanced features such as overspending notification and reminder. We have used the latest mobile technologies to create an application that is robust, secure and compatible with major mobile platforms such as Android and iOS in the nearest future.

## STUDY OF EXISTING SYSTEM

Before proceeding with the conception and realization of our application, we carried out a detailed analysis of the existing system in that domain. This study enabled us to better understand the system such as its functionalities, strength and weakness. Thereby leading us to the following conclusions:

### POPULAR PERSONAL BUDGET MANAGEMENT APPPLICATIONS

We identified many applications already present in the market such as:

* Mint
* YNAB (You Need a Budget)
* PocketGuard

These applications offer the basic functionalities such as tracking expenses, creating budgets, and viewing financial data.

### COMMON FEATURES

Existing applications generally offer features such as:

* Monitoring of income and expenses
* Categorization of transactions
* Creation of personalized budgets
* Financial charts and reports
* Financial planning features

#### USER INTERFACE

Existing applications often focus on intuitive (easy to understand) user interfaces, with clear dashboards, interactive charts, and customization options for spending categories. Some applications offer swiping features to facilitate transaction management.

### LIMITATIONS AND WEAKNESS

Despite the functionalities offered, certain existing applications may present limitations, particularly in terms of:

* Customization of expense categories
* Management of budgets
* Absence of budget alerts settings

All of these contribute to the criticism of the existing system.

### CRITICISM OF THE EXISTING SYSTEM

Criticism makes you evolve,

Table 3:crticism of the existing system

|  |  |  |
| --- | --- | --- |
| CRITICISM | CONSEQUENCES | SOLUTIONS |
| Lack of user-friendliness:  Some existing applications  may be complex to use and lack a user-friendly  interface. | This can make personal budget management difficult for users who are unfamiliar with computer concepts | Set-up of a very user-friendly,  intuitive application enabling  users to easily navigate and  understand budget management  in order to make informed  decisions. |
| Although many applications offer basic features like tracking expenses and creating budgets. They can miss advanced features for deeper financial situation management like  personalization of expense categories or  generation of detailed reports | Restricted budget management  is not adaptive to different users thus leading to it being  abandoned | Implementation of an application  that adapts to the budgetary  management of each user allowing them to better visualize their budget situation |
| Certain applications don’t have the functionality to save data online | Data loss and monitoring of their budgetary situation during the management period. | Implementation of an application  which will allow them to have access to the functionality of storing their data online which will be done online. |
| Lack of budget alert settings | The user is not aware of their overspending | Creation of an application that can give the possibility of configuring alerts. |

Although existing personal budgeting applications are available in the market, there are areas where they can be improved. Better usability, advanced features, deeper data visualization, increased personalization and stronger security are all aspects that could be improved to provide a more complete and efficient user experience in this area. All this therefore leads us to pose a problem.

## PROBLEMATIC

Considering all these shortcomings, how do we design a personal budget management mobile application that offers a user-friendly experience with advanced features, clear data visualization, in-depth customization and enhanced data security, in order to help users to optimize their budget management and archive financial stability?

## SOLUTION PROPOSAL

Noticing the weakness of certain personal budget management applications already present on developed application distribution platforms (Play Store, AppStore), an IT solution taking into account the weakness of certain applications currently available is relevant. The latter will have to address all the weaknesses that they encounter by making its strengths the addition of functionalities.

## CONCLUSION

In conclusion, the analysis of the existing design and production of mobile personal budget management applications revealed several shortcomings and opportunities for improvement. Existing applications may have issues such as limited usability, lack of customization of categories, insufficient data visualization, and data security concerns.

# PART 3: SPECIFICATION BOOK

INTRODUCTION

1. Context and justification of the project
2. The objectives of the project

* General objectives
* Specific objectives

1. Expression of user needs

* Functional needs
* Non-functional needs

1. Project Planning
2. Project cost estimate
3. Project constraints
4. Project planning
5. Deliverables

CONCLUSION

## INTRODUCTION

A specification book is a document which is established by the customer needing a particular product and the producer (engineer) who is to create the product, which is to be respected to its fullest during the period of realization of the product.

## Background and justification for the project

### Context

Effective management of personal finances is a challenge for many people in our society. Between daily expenses, bills, debt, saving goals and investments it can be challenging to keep an overview of your financial situation and make informed decisions. In this context a personal budget management mobile application offers a practical and accessible solution to help individuals better manage their finances.

This application allows users to track their spending, set budgets, view their income and savings, and even set specific goals (e.g. savings).

### JUSTIFICATION

The justification of this theme lies in the many benefits that a personal budget management mobile app can offer and generating reports.

Firstly, it allows you to have a clear view of your financial situation, which makes it easier to make decisions.

Secondly, users can identify areas where they spend excessively and adjust their budgets accordingly, working toward realistic saving goals.

In addition, this application has advanced features such as detailed reporting, financial trend analysis and spending alerts allowing users to better understand their spending habits and identify saving opportunities.

## PROJECT OBJECTIVES

After having identified the context and justified it, it is important for us to give the general and specific objectives of our project.

### General Objectives

We mainly have one main objective which is to provide users with an effective tool to help them manage their personal finances, more specifically their budget, in an optimal manner. The objective therefore encompasses several other specific objectives.

### Specific Objectives

We have several objectives to achieve when implementing our solution such as:

* Expense Tracking and Categorization:

Allow users to set custom budgets for different spending categories, providing tools to track and compare actual spending against fixed budgets.

* Clear visualization of income and expenses:

Provide users with an overview of their income, expenses incurred to help them plan and make informed financial decisions.

* Detailed reporting and analysis:

Provide in-depth reporting and analysis capabilities on spending habits, financial trends and deviation from set budgets, to enable users to better understand their financial situations.

* Alerts and Reminders:

Provide customized alerts and reminders features to notify users of overspending, payment deadlines, and savings goals.

* Data Security:

Ensure the security and privacy of users’ financial data by implementing robust protection measures, such as data encryption and secure authentication.

* User-friendly user interface:

Design intuitive, friendly and attractive user interfaces, with smooth navigation and easily accessible features, to provide a pleasant and engaging user experience.

### Expression of user needs

The analysis, design and creation of an application is based on very specific needs which can be functional or non-functional

1. Functional needs.

A functional need refers to a specific functionality that a system or product must offer to meet customer expectations. These needs focus on the actions and tasks the user wants to accomplish with the system such as:

* Categorization of expenses and income:

Offer the possibility of classifying expense and income into different categories (food, housing, transport, etc.) for better visualization and analysis.

* Budget creation:

Allows users to define personalized budgets for different spending categories and set monthly spending limits.

* Progress tracking:

Provide visual graphs to allow users to track their progress against their budget and savings goals.

* Overspending alerts:

Send notifications to users when their spending exceeds a predefined threshold to help them control their spending.

* Financial Reports:

Generate detailed reports on expenses, income, savings and financial trends for in-depth analysis.

* Management of user accounts:

This means that the administrator, through his actions, will be able to delete and have an overall view of the use of the application by users who are not administrators.

## Non-functional needs

A non-functional need, also called a non-functional requirement, concerns features and constraints that go beyond the specific functionality of the system. These needs focus on aspects such as performance, security, accessibility, usability, reliability, compatibility, availability etc.

* Performance

The application must be able to provide rapid response to user interactions, and handle complex calculations related to budgets, reporting, and analytics.

* Data security:

Users’ personal financial information must be secure and protected from unauthorized access, using appropriate encryption, authentication and backup measures.

* User and accessibility:

The user interface of the application should be user-friendly, with a clear layout, intuitive icons and easy navigation to make it easy for users.

* Reliability:

This application must be stable, without frequent crashes, and it must be able to correctly recover from errors or connection failures.

* Cross-platform compatibility:

The application should be compatible with different popular mobile operating system, such as Android and iOS, to allow users to use it on a variety of devices.

* Availability:

They should be available online or offline, allowing users to access their data and manage their budget even in the absence of internet connectivity.

* Scalability:

The application must be designed to handle increasing volumes of data and be able to adapt as new features are added or to support an increasing number of users.

* Privacy:

The application must comply with personal data protection regulations and provide users with control over the collection, use and the collection, use and disclosure of their financial information.

* Support and maintenance:

The application should be supported by a development team that provides regular updates, fixes bugs, and responds to user support requests.

## Project Planning

Generally, in project management, planning is essential to good management. To do this, we used the Gantt Project software to explain in the form of a graph how our project evolved over time, from the analysis to implementation through design.

### GANTT PROJECT

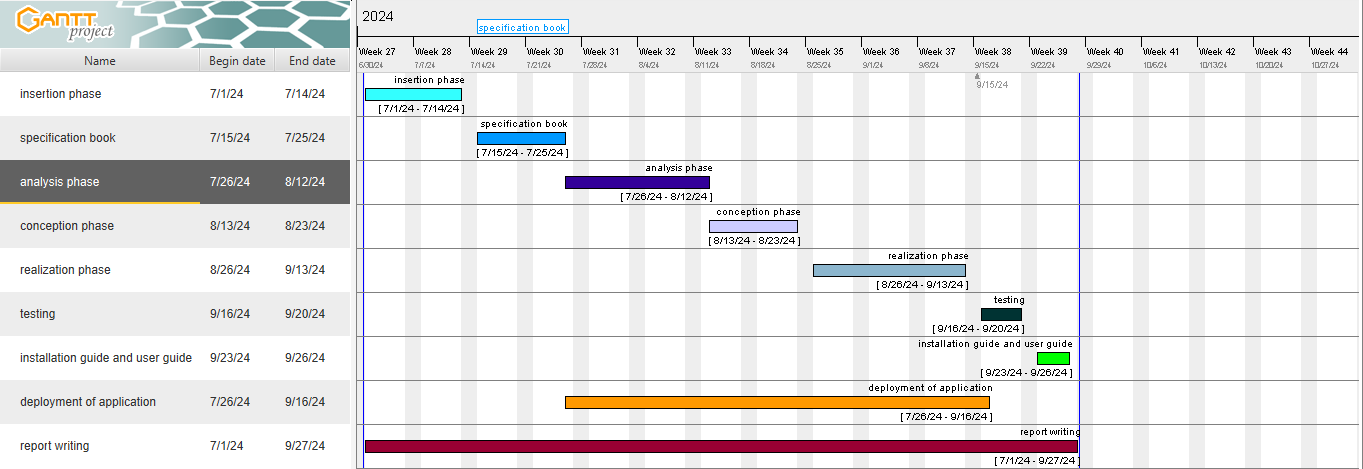
****

Figure 3:Gantt project

### ESTIMATED COST OF THE PROJECT

#### Human resource

The table below consist of the necessary human resources:

Table 4:human resource (MERCURIAL 2023)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Resources** | **Quantity** | **Role** | **Cost Per Day (FCFA)** | **Number of Days** | **Cost (FCFA)** |
| Analyst Designer | 01 | He will carry out the work of analyzing the problem and purpose | 150000FCFA | 18 | 2700000FCFA |
| Developer | 01 | His role is to produce an application on the basis of analysis produced by the analysts. | 100000FCFA | 28 | 2800000FCFA |
| Testers | 01 | He will carry out the unit tests in order to validate the solution obtained | 50000FCFA | 8 | 400000FCFA |
| Designer | 01 | He will be in charge of designing the different interfaces of our application | 100000FCFA | 10 | 1400000FCFA |
| TOTAL |  |  |  |  | **7300000FCFA** |

**Source:** Mercurial 2023

#### Software resources

Table 5:software resource (MERCURIAL 2023)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Resources** | **Software** | **Usage** | **Quantity** | **Price (FCFA)** |
| **Database Management system** | Mongo DB | Database communication | 01 | free |
| **Operating System** | Windows 10 Pro | Computer’s operating system | 01 | 119,944 |
| **Development tools** | * Visual studio code * Node js * React | * Integrated development environment for code editing * Backend development * Frontend development | 01 | free |
| **Project Planner** | Gantt Project | Project planning | 01 | free |
| **Word processor** | WORD 2019 | Realization of internship report | 01 | 287 500 |
| **Modelling Tool** | Visual Paradigm | Modelling of system | 01 | free |
| **TOTAL 2** |  |  |  | **407,444** |

#### Hardware resources

The table summarizes the equipment essential for deploying the application.

Table 6: Hardware resources (MERCURIAL 2024)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Resource** | **Hardware** | **Usage** | **Quantity** | **Price (FCFA)** |
| **Computer** | DELL windows 10 | Documenting, Analysis and coding | 1 | 450000 |
| **Installation media** | CD-ROM | Installation media | 1 | 500 |
| **Removable disk** | 64gb USB Key | For file transfer and backup | 1 | 3000 |
|  |  |  |  |  |
| **TOTAL 3** |  |  |  | **453 500** |

#### Total project cost

The table below summarizes the total cost required to deploy the application.

Table 7:total project cost

|  |  |
| --- | --- |
| Resources | Prices (FCFA) |
| Human resources | **7,300,000** |
| Hardware resources | **453,500** |
| Software resources | **407,444** |
| Unexpected (10%) | **300,000** |
| Total | **8,155,944** |

## Project Constraints

A constraint is a condition to be respected by the stockholders for the smooth running of the project. To successfully complete this project, the following constraints apply:

* Cost:

We must respect the fixed price in the financial study and avoid overestimation or underestimation of this price estimated at

* Deadline:

The project must respect the time interval defined by mutual agreement by the project managers and project owners which must extend over a period from July 1 2024 to September 30 2024.

* Quality:

After the price and a fixed deadline, we must produce a good quality application that meets the expressed needs and should allow easy use for users. The application must be very efficient, extensible and scalable

## PROJECT TEAM AND DELIVERABLES

### Project Team

Table 8: Project team

|  |  |
| --- | --- |
| **NAMES** | **TASKS AND ROLES** |
| **EPAH FONZOCK MANYI** | **Project head, analyst, realizer and coder of project** |
| **Mr. Messio** | **Lecturer at AICS (Academic supervisor)** |
| **Mr. NGUH Prince** | **Software Engineer and entrepreneur (Professional supervisor)** |

### Deliverables

The project will be only be received if at the end of the internship period we provide the following deliverables:

* A prototype of the application burned on a DVD.
* Specification
* An analysis file
* A design file
* A production file
* A functionality testing file
* A user guide and installation guide

## CONCLUSION

In conclusion, the specifications for a personal budget management mobile application must take into account both functional and non-functional needs. Well-defined specifications will make it possible to create a high performance, secure and user-friendly personal budget management application, offering users a smooth experience and allowing them to effectively manage their finances.

# PART THREE: ANALYSIS PHASE

# INTRODUCTION

The analysis document which follows the specification book, allows us to present a detailed analysis of the problem and the solution, the chosen analysis method and the reasons behind this choice. To achieve this, we will use UML 2.5 (Unified Modelling Language) alongside 2TUP (2 Track Unified Process) as method applied to UML to analyze the system.

INTRODUCTION

1. PRESENTATION OF THE MODELLING APPROACH
2. JUSTIFICATION OF THE ANALYSIS METHOD
3. RISK ANALYSIS OF THE PROJECT
4. MODELLING OF THE SYSTEM

CONCLUSION

## INTRODUCTION

The analysis of a project is above all, the most delicate stage during the realization of the project. The analysis phase is where you break down the deliverables of a high-level project. System analysis is essentially a viewpoint for defining problems, developing alternative solutions and evaluating these solutions. It is in this phase that details of the project mentioned in the specification book are explicitly explained. We show which method and processes are being used, why these methods are being used over another and how they are being adapted to the business rules of the studied environment.

## PRESENTATION OF THE ANALYSIS METHOD

### COMPARATIVE STUDY BETWEEN MERISE AND UML

**UML** was first founded by Grady Booch, Ivar Jacobson, and James Rumbaugh in 1995 while working at Rational Software.

UML 1.x was the first series of UML versions, releases between 1997 and 2005. It consisted of 13 Diagram types, organized into four categories; Structural, behavioral, interaction, and use case.

**MERISE** first version was introduced in 1978-1979 with the aid of several branches of the French government. By the mid 1980’s, it became a standard in France and other countries in financial information system development.

MERISE is a general-purpose modeling methodology in the field of information systems development, software Engineering and project management.

#### DIFFERENCES BETWEEN MERISE AND UML

Table 9: Differences Between MERISE and UML

|  |  |
| --- | --- |
| MERISE | UML |
| This is a method of analysis | This is a modeling Language |
| Stands for “Methode d’Etude et de Realisation Informatique pour les Systemes d’Entreprises.” | Stands for “Unified Modeling Language” |
| Does not require any process attached to it | Coupled with 2TUP to form a method on its own. |
| MERISE can only design database models | UML mainly used object-oriented design |

### 

### COMPARATIVE STUDY BETWEEN 2TUP AND UP

Table 10:differences between 2tup and up

|  |  |
| --- | --- |
| **2TUP** | **UP** |
| The 2-track unified process is based upon a SPEM modeling architecture in order to conceive elegant and adapted solutions but also to take advantage of a few techniques and technologies. | The Unified process (UP) or Unified Process Model (UPM) is an iterative, incremental, architecture-centric, and use-case driven approach to software development. |

#### CHOICE OF METHOD

* **Process**

A process in system development is defined as a series of steps or activities, partially ordered which permits to obtain software system or evolution of an existing software.

* **Unified process**

A unified process is an extensible process framework of development of software built on UML.

It is iterative, incremental, centered on architecture, driven by use cases and driven by risks.

* Iterative: The method is iterative since it allows to do iterations during its different phases thus guarantying that the model produced at each phase is refined and ameliorated.
* Incremental: Incremental here depicts the fact that each iteration is supposed to offer incremental improvement over the previous iterations enlarging the system to its completion.
* Centered on architecture: The different models defined during the process of development contribute to the establishment of a coherent and solid architecture.
* Driven by use cases: That is to be centered on users’ needs and satisfaction.
* Driven by risk: Enables to define priorities for each functionality that minimizes risks of project failure.

## 

## THE TWO TRACK UNIFIED PROCESS

2TUP is a unified process which has as objective to bring a solution to constraints of functional and technical changes imposed on information systems. It proposes a development cycle which separates the functional aspects from the technical aspects. It goes from the point of view that all evolution imposed on the system could be decomposed and treated in parallel following the functional branch and the technical branch. 2TUP distinguishes therefore two branches (the functional and technical branch) which their different result is combined to realize our system. After the evolution of the functional and the technical branch, the realization merges the results of the two branches. The merging permits to obtain a process of development in the form of Y as shown in the figure below.

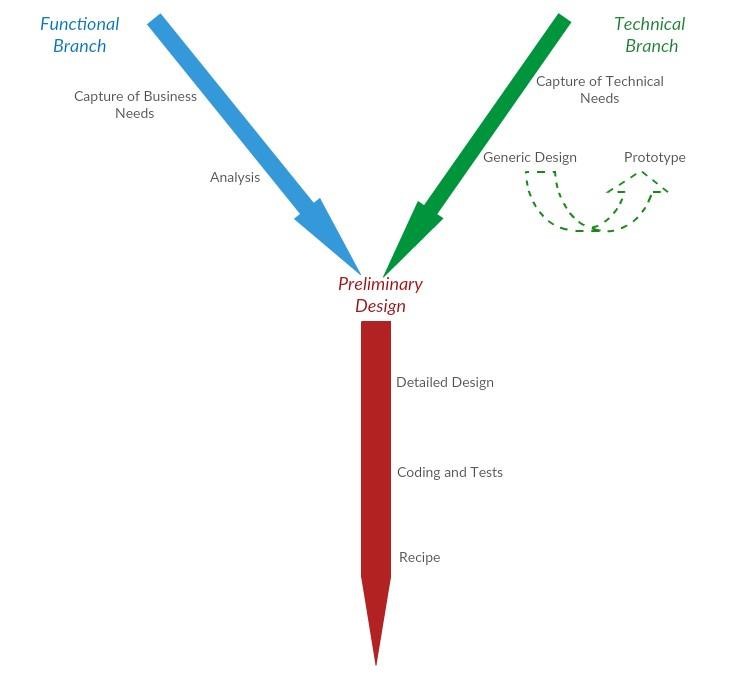


Figure 4:Track Unified Process (Source:Google)

### Functional Branch

The functional branch captures functional needs of the system and how it behaves. This phase specifies the elements of the preliminary study and does not depend on technology used to build the system.

### Technical Branch

The technical branch enumerates the technical needs and purpose a generic design validated by a prototype. The technical needs here include the tools, materials, and technologies that will be used along with the different constraints such as worst-case scenario and integration with the existing controls.

### Realization Branch

The realization track supports the following:

* The Preliminary Conception: This is the most sensitive step of the 2TUP. This is the meeting point between the functional and technical branches. It ends when the deployment model, the operating model, the logical model, interfaces and software configuration model are defined;
* The detailed conception: This is the detailed design of each feature of the system;
* Coding and testing: This is the programming phase of the designed features, alongside testing of the code features;
* The Recipe: This is the validation phase of the functions of the system developed.

## 

## MODELLING WITH UML 2.5

The 2TUP and UML work hand-in-gloves. Having already discussed briefly on 2TUP, UML being its foundation is equally as important. UML stands for Unified Modelling Language. It can be used to model a system independent of the platform language. UML is a graphical language for visualizing, specifying, constructing, and documenting information about software-intensive systems, some of which bare object oriented. UML gives a standard way to wrote a system model, covering conceptual ideas, hence it is a privilege vector of communication between members of a team. Before proceeding, here are some key terms to understand;

Model: A model is an idealized, abstract and simplified representation of a real-world object or simplified simulation of an entity.

Why is UML unified? UML is said to be unified because it comprises of three modelling approaches namely:

* Object Modelling Technique (OMT): which provides a graphical representation of the static, dynamic and functional aspects of the system;
* Brooch approach: Which is excellent for design and implementation. It introduces the notion of packages;
* OOSE (Object-Oriented Software Engineering): An approach which focuses on design based on the users’ needs.

How is UML a Language? The UML notations are a standard that is widely used in the professional milieu. The notations are a must, however the usage of these notations in a software development approach are not, hence it is just a guideline.

UML 2.5 defines 14 diagrams that are classified into two main categories; Structural and Behavioral diagrams.

* + - 1. STRUCTURAL DIAGRAMS

Structural diagrams show the static structure of the system and its parts on different abstraction and implementation levels and how they are related to each other. The elements in a structure diagram represent the meaningful concepts of a system, and may include abstract, real and implementation concepts. In UML 2.5, the different structural diagrams include;

* Class Diagram;
* Communication Diagram;
* Deployment Diagram;
* Object Diagram;
* Package Diagram

### BEHAVIORAL DIAGRAMS

Behavioral diagrams show the dynamic behavior of the objects in a system from the beginning of a task to its completion. Below are some behavioral diagrams that exist in UML 2.5;

* Use case diagrams;
* Sequence Diagrams;
* Activity Diagrams.

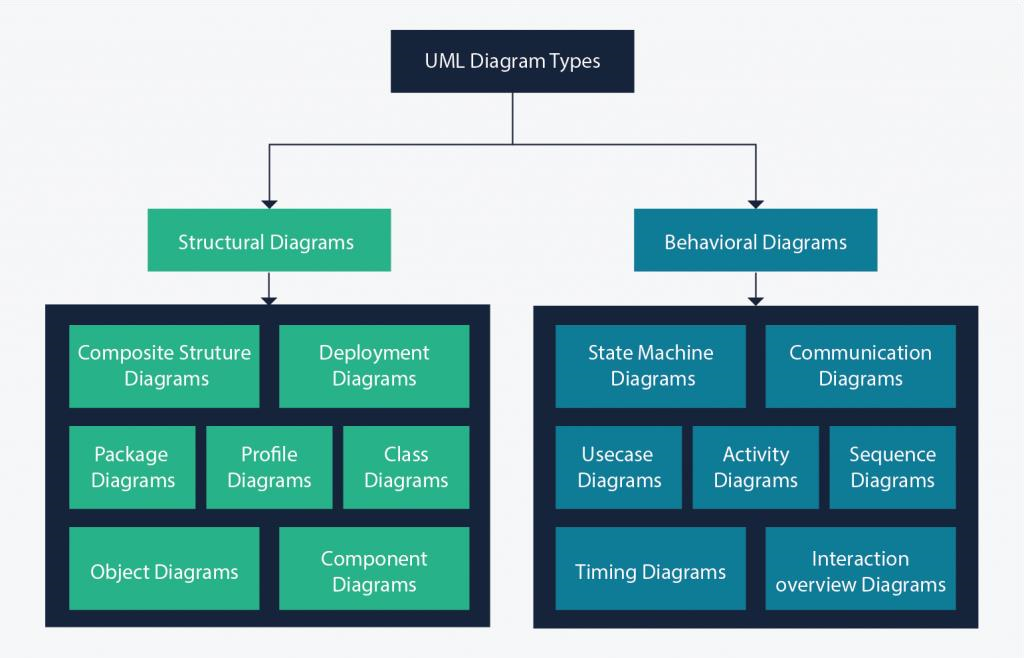


Figure 5:UML Diagram (Source Google)

## 

## JUSTIFICATION OF THE ANALYSIS METHOD

Our choice for the UML modelling language and the 2TUP development process are based on the following criteria:

* UML is a language which is centered on the user’s needs;
* UML is based on the object approach;
* UML produces good standards for software development;
* UML has large visual elements to construct and easy to follow;
* 2TUP is a process based on object approach and is constructed on UML
* 2TUP facilitates the modelling of complex systems which will undergo many evolutions with time;
* 2TUP offers a development cycle (the Y shape development cycle) which dissociate the technical aspects from the functional aspects
* 2TUP is a unified process, so it possesses all the characteristics of this process

### 

## RISK ANALYSIS OF THE PROJECT

A software project can be exposed to serious potential risks that can ruin the whole project. This is why it is very important to identify and try to migrate the potential risks that our project may encounter. Risk analysis is defined as the process of identifying and analyzing potential issues that could negatively affect a project in order to help project’s stakeholders avoid or mitigate those risks.

For our project which is software oriented, we could identify and classify the potential risk that may threaten our system into different categories. Below is a table of these risks, their description, and the response that will be taken for each risk if it materializes.

Table 11: Risk analysis of the project

|  |  |  |
| --- | --- | --- |
| Risk and description | Causes | Solution |
| Budget Risk: This is when the required investment is inaccurately anticipated. | Inaccurate Cost Estimation: Certain required items excluded from the estimation of costs | Incremental review of cost planning and budget provision for unforeseen expenses |
| Operational Risk: Ineffective processing, system failures, or unanticipated circumstances define operational risk, this risk reduces productivity | * Failure to establish testing priorities * Improper communication between team members * Improper communication with administrator | * Team should prioritize key functionalities of the system and work in short iterations * Plan and foster prompt communication and project team stakeholders * Provide coaching and team development. |
| Technical Risk: This type of risk leads to professional and performance failure | * Product complexities * Continually changing requirement | Model the system in order to visualize and manage complexities first on early specifications and plan added specification as new project or iteration |

### 

## MODELLING OF THE SYSTEM

### CAPTURE OF THE FUNCTIONAL NEEDS

The functional requirements of a system specify the desired behavior or basic functionalities the system is required to perform. In this section, we are going to present the following diagrams;

* Use case diagram and textual description of the detailed use cases;
* Activity Diagram.

#### Use case diagram

The use case diagram simply shows the functionalities of the system, their interdependency and how they are links with actors of the system.it also clearly defines the boundaries of the system and it’s the first diagram to construct.it is at this stage that the analysis phase of a system begins

**FORMALISM**



Figure 6:formalism of a use case diagram

Table 12: Components of a use case diagram

|  |  |  |
| --- | --- | --- |
| Element | Description and Main properties | Notation |
| **Actor** | Represents an entity that directly interacts with the system. The actor is what performs the different possible actions of the system. |  |
| **Use case** | A use case represents a functionality of the system. It is an action that can be performed by an actor. |  |
| **Association** | It indicates that an actor takes part in a use case. |  |
| **Inclusion** | An inclusion relation denotes that an included action must be performed before another action can be performed. A includes B signifies that B is a compulsory part of A. s |  |
| **Extension** | It denotes that an action may be performed while another one is being performed. A use case B extends A Means B is an optional part of A. |  |
| **Inheritance** | It is the only possible relation between actors. |  |
| **Generalization** | This shows that a uses case is a kind of another. This relation also permits to decompose a complex case into smaller and simple cases. Abstract and concrete actors can be defined and later specialized using Generalization relationship. |  |

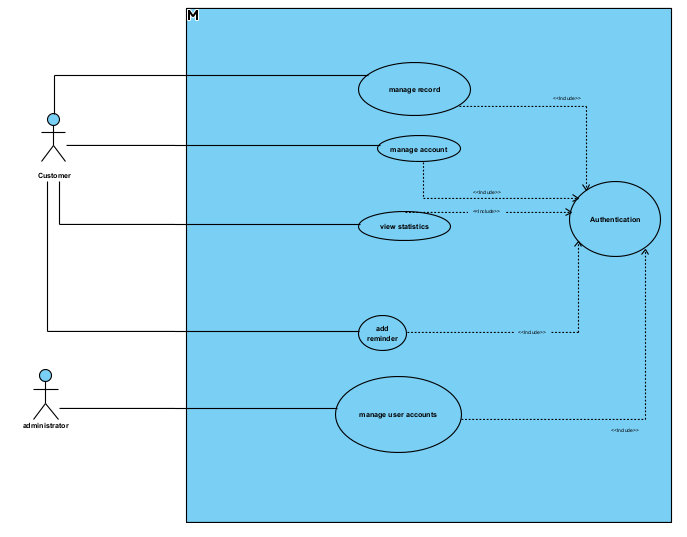


Figure 7:use case diagram

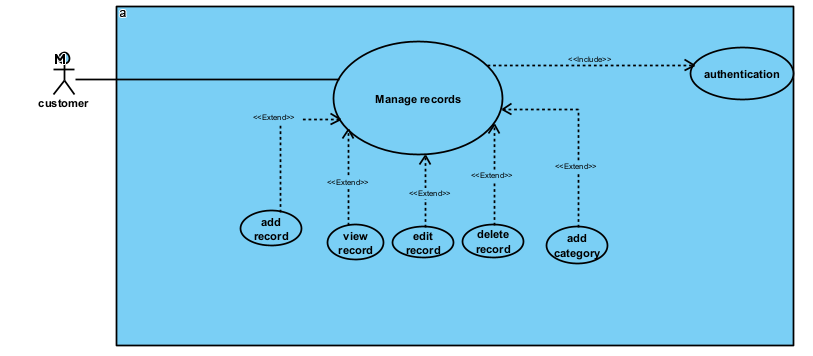


Figure 8:specific use case diagram for manage records

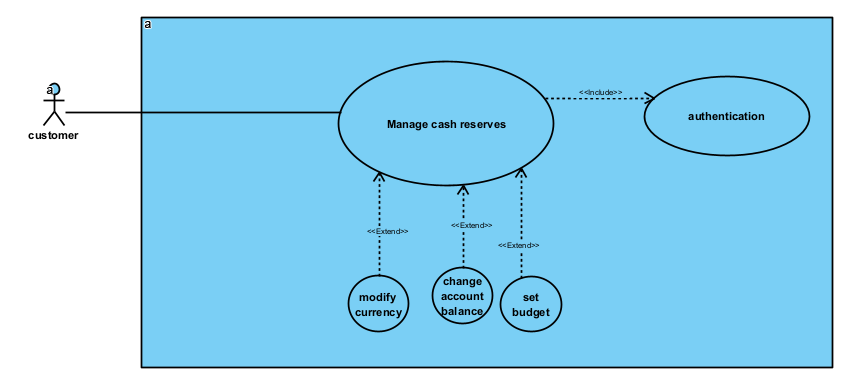


Figure 9:specific use case diagram for manage cash reserves

Textual description of use case <<Create account>>

Table 13: Textual description of use case create Expense

|  |  |
| --- | --- |
| **Name Create expense** | |
| **Objective** | To allow user to create an expense |
| **Actors** | Client |
| **Date** | 07/20/2024. |
| **Stakeholders** | Admin. |
| **Presumption** | The application has already been launched. |
| **Pre-condition** | The user already has an account. |
| **Normal scenario** | 1. User clicks on login button. 2. System displays the dashboard. 3. The user clicks on the record navigation bar. 4. The user clicks on the add record button. 5. The fills the form and submits the expense form. 6. The system verifies the conformity (if data is correctly entered) 7. The system displays success message |
| **Alternative scenario** | 1. At step 5 of the normal scenario, in case of form conformity error, the system displays an error message 2. System doesn’t display any added record |
| **Post condition of failure** | An error message is displayed and the system doesn’t add any records |

Table 14:textual description of authentication

|  |  |
| --- | --- |
| **Name** | **AUTHENTICATE** |
| **Objective** | The use case is to permits users to log in to their workspaces |
| **Actor** | All users |
| **Date** | 07/20/2023 |
| **Stakeholder** | Admin |
| **Pre-condition** | User must be a member of the system |
| **Trigger** | User clicks on the button login |
| **Nominal Scenario** | 1. The system displays the fill login form 2. User enters credentials and sends 3. The system checks if credentials are correctly filled. 4. System displays success message 5. System displays work space |
| **Alternative Scenario** | 1. At step 3, if credentials are wrongly entered the system will display an error message 2. If the information doesn’t exist, the system displays an error message. |

### Activity Diagram

In UML an activity diagram is used to display a sequence of activities. Activity diagrams show the workflow from start point to the finish point detailing the many decision paths that exist in the progression events contained in the activity. They may be used to detail situations where parallel processing may occur in the execution of some activities. Activity diagrams are useful for business modelling where they are used for detailing the processes involved in business activities.

**FORMALISM **

Figure 10:formalism of an activity diagram

Table 15:components of an activity diagram

|  |  |  |
| --- | --- | --- |
| **Element** | **Description and Main Properties** | **Notation** |
| Action state | An Action state represent the non-interruptible action of an object | Action |
| Activity | An Activity encloses all actions, control flows and other elements that make up an activity. | Activity |
| Start point or Initial State | This represents the initial action state or the start point of an activity diagram usually placed at the top left corner of the first column |  |
| Action Flow | Action flow illustrate the transition from one action state to another |  |
| Decision Branch | A Decision Branch is added between two activities when a decision is needed prior to move to a latter activity | Decision |
| Final State or End Point | The Final State or End Point represent the final action state of an activity |  |
| Flow Final Node | Indicates the end of a single flow within the activity. |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **Merge Point** |  | A merge node marks the end of the conditional behavior started at the decision node. Merges are also shown with diamond shaped nodes, but they have multiple incoming edges and one outgoing edge. |  |
| **Fork Node** |  | Indicates steps that occur at the same time or that are said to occur concurrently or in parallel. |  |
| **Join Node** |  | The join means that all incoming actions must finish before the flow can proceed pass the join. |  |
| **Send and**  **Signal Node** | **Accept** | The send signal node shows that a signal is sent to an outside participant and the receive signal node shows that the signal is received from an external process. |  |
| **Final State or End**  **Point** | | An arrow pointed to a filled circle nested inside another circle represents the final action state of an activity diagram. |  |

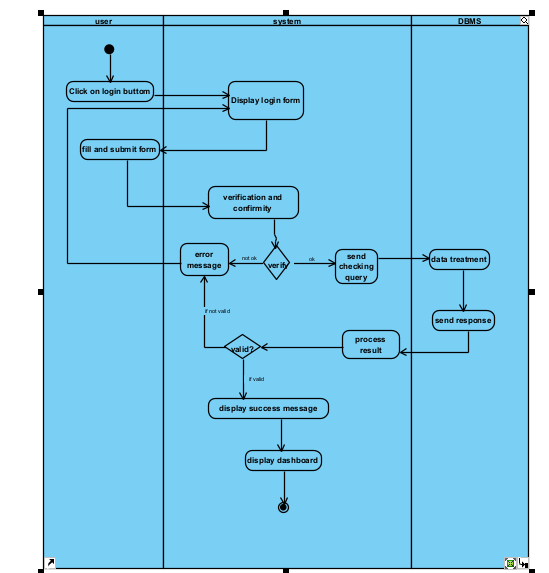


Figure 11:activity diagram for authentication

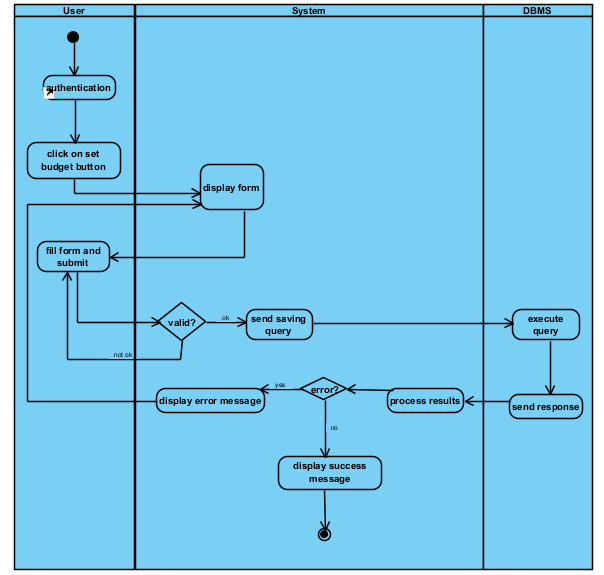


Figure 12:activity diagram for set budget

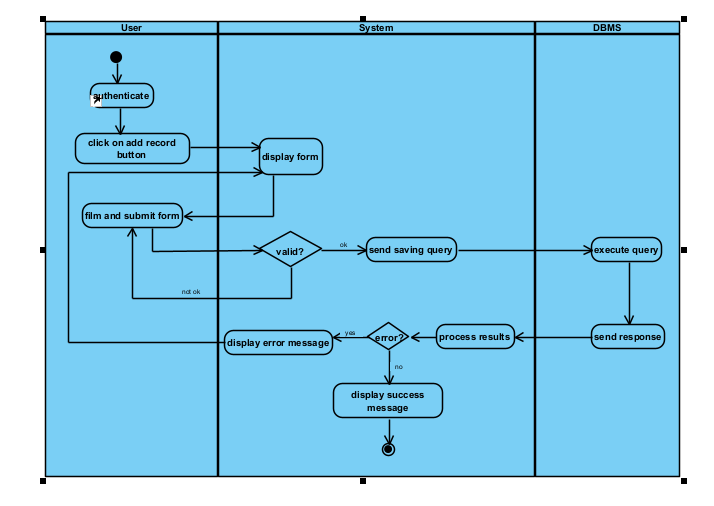


Figure 13:activity diagram for add record

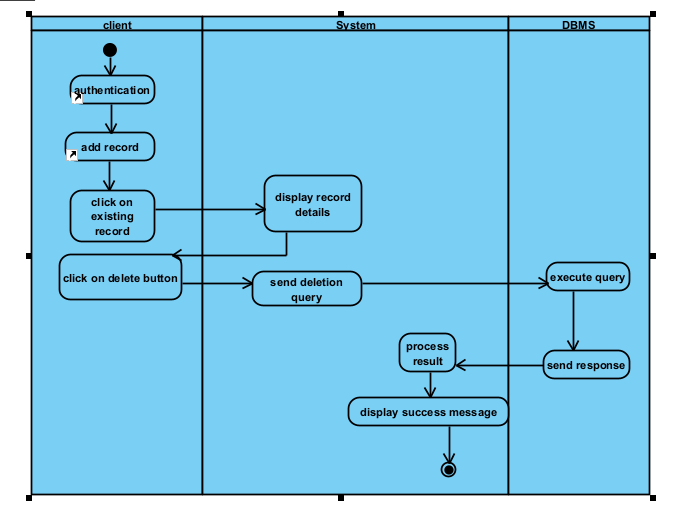


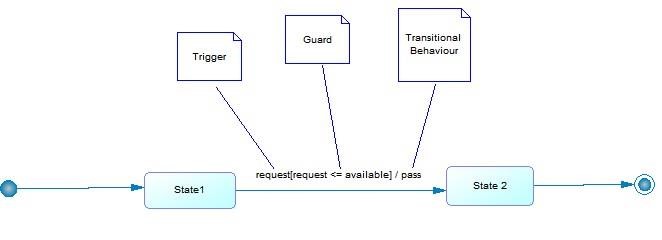
Figure 14:activity diagram for record deletion

1. **STATE MACHINE DIAGRAM**

A State Machine Diagram describes the behavior of a single object in respect to a series of event in a system. Also known as the State Chart diagram, it models the dynamic flow of control from state to state of a particular object within the system.

**FORMALISM**

Figure 15:formalism of a state machine diagram



#### Fig 18: Formalism of a state machine Diagram

Table 16: Components of state machine Diagram

|  |  |  |
| --- | --- | --- |
| **Element** | **Description and Main properties** | **Notation** |
| State | A State represents situations during the life of an object. | State |
| Transition | A solid arrow that represents the path between different states of an object. A state can have a transition that points back to itself. |  |
| Initial State | A filled circle followed by an arrow represents an object’s initial state. |  |
| Final State | An arrow pointed to a filled circle nested inside another circle represents an object’s final state. |  |
| Join | It is used to split a transition or reduce two or more transitions into one |  |

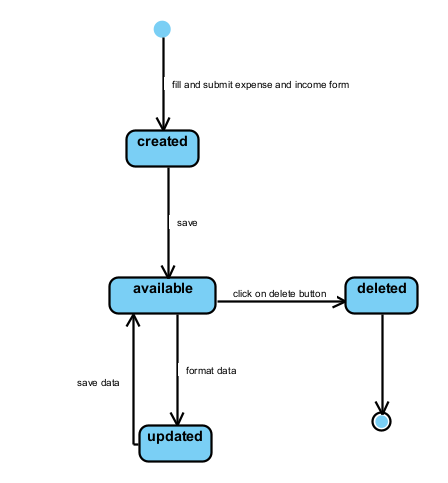


Figure 16:state machine diagram for record

1. **Sequence diagram**

Sequence diagram shows a detailed flow for a specific use case or even just part of a specific use case. They are almost self-explanatory; sequence diagrams are good at showing which objects communicate with other objects and what messages trigger those communications.



Figure 17:formalism of a sequence diagram

Components of a sequence diagram

Table 17: Components of sequence Diagram

|  |  |  |
| --- | --- | --- |
| **Element** | **Description and Main properties** | **Notation** |
| **Life lines** | A lifeline represents an individual participant in a sequence diagram. A lifeline will usually have a rectangle containing its object name as shown in the diagram below; |  |
| **Asynchronous message** | Messages sent with no feedback |  |
| **Synchronous message** | A message with complete feedback |  |
| **Self - message** | A self-message can represent a recursive call of an operation, or one method calling another method belonging to the same object. |  |
| **Return message** | Return messages are results of asynchronous messages |  |
| **Object** | They send and receive messages |  |
| **Fragment** | It is the general relationship type between elements. This connector may include named roles at each end, cardinality, direction and attributes. |  |

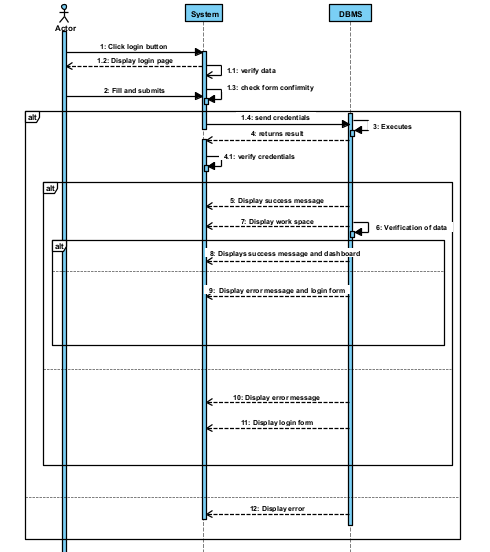


Figure 18:sequence diagram for authentication

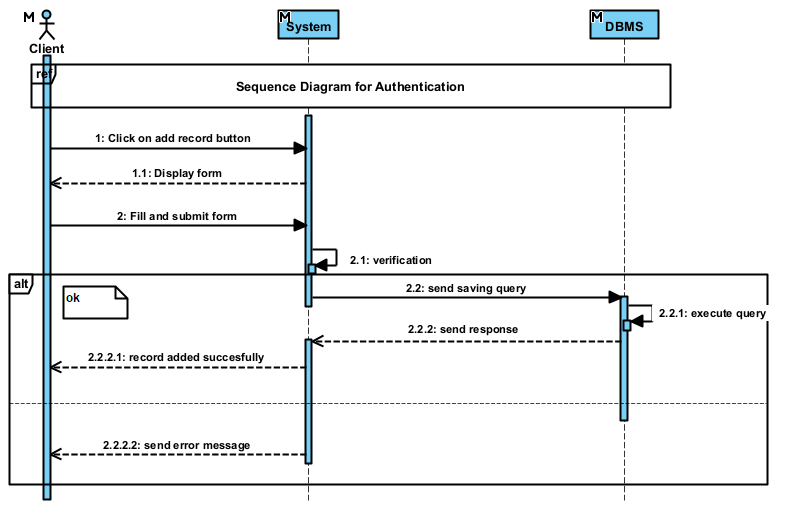


Figure 19:sequence diagram for add records

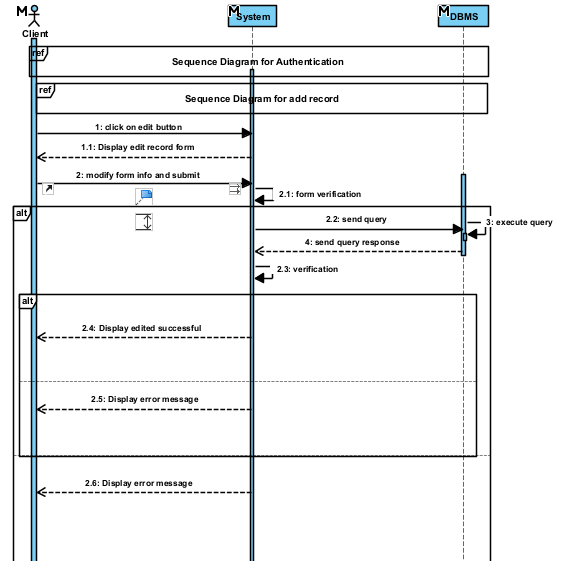


Figure 20:sequence diagram for edit records

1. **COMMUNICATION DIAGRAM**

A communication is a type of interaction diagram which shows how objects interact. A communication diagram is an extension of a class diagram that show the objects along with the messages that travel from one object to another. Communication diagram also show the messages the objects send to each other.

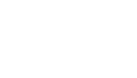


Objet\_1



1:

Message\_1



2:

Message\_2



3:

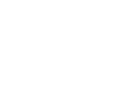
Message\_3



Objet\_4



Objet\_3



Objet\_2

Figure 21:formalism of a communication diagram

COMPONENTS OF A COMMUNICATION DIAGRAM

Table 18: Components of communication diagram

|  |  |  |
| --- | --- | --- |
| **Element** | **Description and Main properties** | **Notation** |
| **Call message** | A call message defines a particular communication between lifelines of interaction that represents an innovation of target lifeline. |  |
| **Dependency** | A dependency is a relationship that signifies that a single or set if model elements require other model elements for their specification. |  |
| **Object** | An object represents an individual participant in the interaction conversion |  |
| **Generalization** | A generalization is a taxonomic relationship between a more general classifier and a more specific classifier |  |

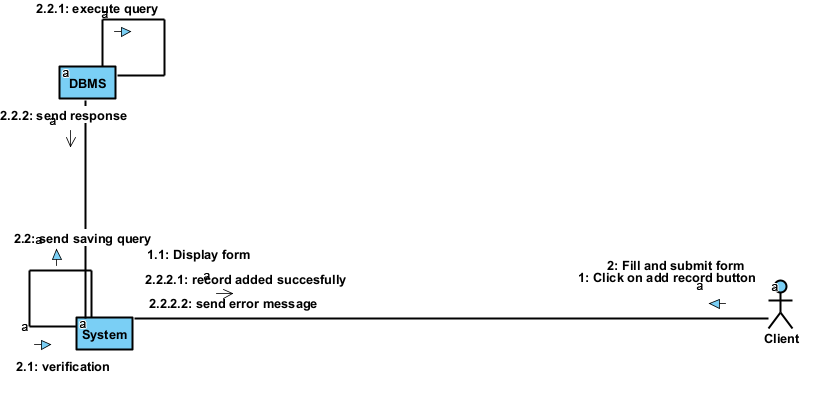


Figure 22:communication diagram for add records

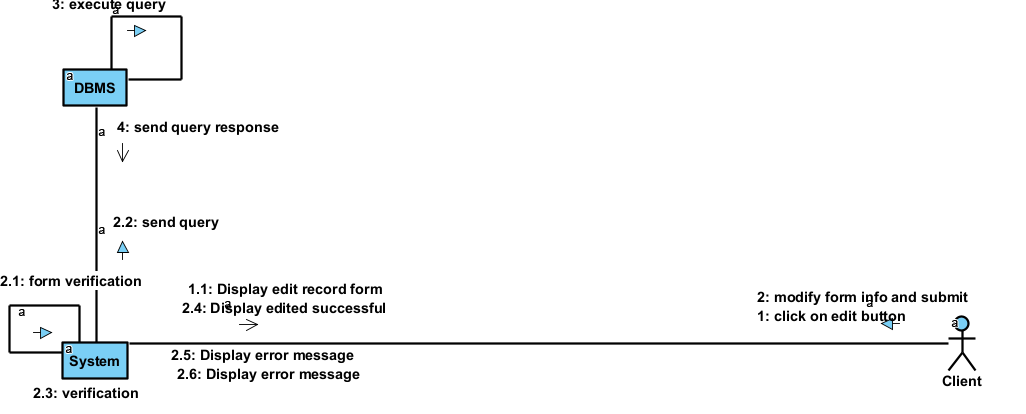


Figure 23:communication diagram for edit records

CONCLUSION

With the analysis carried out, we now have a detailed and clear understanding of the functional needs of the system we are to implement. We will now move on to the conception phase where we will see the detailed conception of the system. We will capture the technical needs needed in the realization branch

# PART FOUR: CONCEPTION PHASE

## INTRODUCTION

The conception phase describes, in detail, the necessary specifications, features and operations that will satisfy the functional requirements of the proposed system as modelled in the analysis phase. This phase is meant to identify and consider the essential components (hardware and/or software), structure (networking capabilities), processes and procedures for the system to accomplish its objectives. Therefore, we shall see in this phase the technical needs of the system, the generic conception, the preliminary conception and the detail conception of our application.

##### 

##### TECHNICAL BRANCH

### GENERIC DESIGN

The generic conception consists of developing a solution that responds to the technical specification. This conception is independent of the functional aspect specified in the functional branch of the 2TUP. THE PHYSICAL ARCHITECTURE being a web platform, our system requires a specific client/server system where a user requests a resource, and the server provides it directly or indirectly. A computer architecture is structured in 3 layers, namely; the presentation layer, the treatment layer, and the data layer. The kernel of an application consists of the presentation logic and the processing logic, and it is the distribution of the core between the different layers that determines the architecture characteristics of the client/server systems, we can mention; The 2-tier, 3-tier, n-tier. Our platform will be deployed on a 3-tier architecture. This architecture can be described by the following figure

Figure 24:Three-tier architecture (source: Softwaretestingclass.com)

A 3-tier architecture is a type of software architecture which is composed of three “tiers” or “layers” of logical computing. They are often used in applications as a specific type of client server system. Three-tier architectures provide many benefits for production and development environments by modularizing the user interface, business logic, and data storage layers. Doing so gives greater flexibility to development teams by allowing them to update a specific part of an application independently of other parts.

#### Presentation tier

The presentation tier is the front-end layer in the 3-tier system and consists of a user interface. The user interface is often a graphical one accessible through a web browser or web-based application and which displays content and information useful to and end user. This tier is often built on web technologies such as HTML5, JavaScript, CSS, or through other popular web development framework, and communication with other layers through API calls.

#### Application tier

The application tier contains the functional business logic which drives an application’s core capabilities. It is often written in Java, NET, C#, Python, C++, etc.

#### Data Tier

The data tier comprises the database/data storage systems and data access layer. Examples of such systems are MySQL, Oracle, PostgreSQL, Microsoft SQL, MongoDB, etc. Data is accessed by the application layers via API calls.

### LOGICAL ARCHITECTURE

Taking into consideration that the platform has to be maintainable, the model we have chosen for the logical architecture of our system is the MVC (Model View Controller). It is a software design pattern for developing web applications and web pages. In this part, we had as essential objective to show how our platform is going to be deployed, the use of tools such bootstrap as framework of CSS (Cascading Style sheet), XAMPP as web server and others that enable us to implement our project. Now that our solution has been put in place, we can produce an installation guide and a user guide.

* **Model:** The lowest level of pattern which is responsible for maintaining data.
* **View:** This is responsible for displaying all or a portion of data to a user.
* **Controller:** Software code that controls all the interactions between the model and the view

## IMPLEMENTATION BRANCH

Here, we will see the preliminary design, detailed design and documentation of the system.

### Preliminary design

#### Package Diagram

A package is a namespace used to group together elements that are semantically related and might change together. It is a general-purpose mechanism to organize elements into groups to provide better structure for system model. Owned members of a package should all be packageable elements. If a package is removed from a model, so are all the elements owned by the package. Package by itself is a package able element, so any package could also be a member of other packages.

Formalism of a package diagram

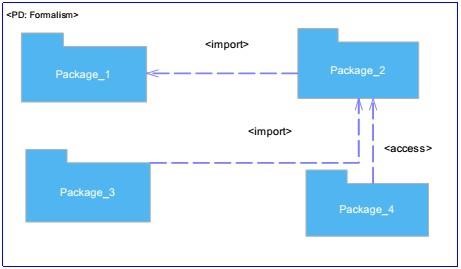


Figure 25:formalism of a package diagram

Components of a package diagram

Table 19: Components of a package diagram

|  |  |  |
| --- | --- | --- |
| Element | Diagrammatical Representation | Description |
| Package | |  | | --- | | package | | It is a namespace used to group together elements that are semantically related and might change together. It is a general-purpose mechanism to organize elements into groups to provide better structure for system model. |
| Dependency | ---------> | A relationship between two models where a change in one will similarly affect the other |

#### Deployment diagram

The deployment diagram shows how a system will be deployed in the hardware environment. Its purpose is to show where the different components of the system will physically run and how they will communicate with each other.

**Formalism of a deployment diagram**

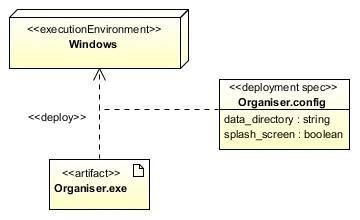


Figure 26:Formalism of a deployment diagram

Components of a deployment diagram

Table 20: Components of Deployment diagram

|  |  |  |
| --- | --- | --- |
| **Element** **Description and Main properties** | | **Notation** |
| **Node** | A node is either a hardware or a software component | Node\_5 |
| **Artifact** | An artifact is a product of the software development process. | a artifact |
|  |  |  |
| **Component** | It represents a modular part of a system that encapsulates it content and whose manifestation is replaceable within its environment | ComponentInstance\_1 |
| **Association** | An association represents a communication path between nodes. |  |

#### Component diagram

Formalism of a component diagram



Composant\_1



Composant\_2



Composant\_3



Composant\_4



Interface\_1



Composant\_5

Figure 27:formalism of a component diagram

### Detailed Design

#### Class Diagram

A class diagram in the Unified Modelling Language is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among objects.

In the diagram, classes are represented with boxes that contain three compartments:

* The top compartment contains the name of the class.
* The middle compartment contains the attributes of the class.
* The bottom compartment contains the operations the class can execute.

Formalism of a class diagram

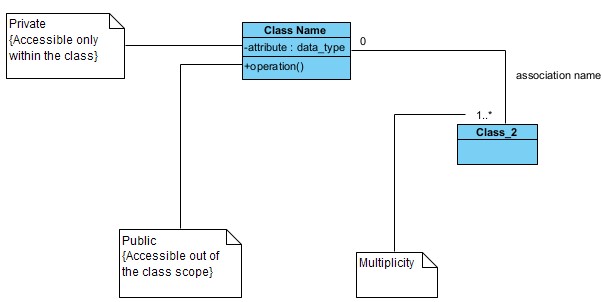


Figure 28:formalism of a class diagram

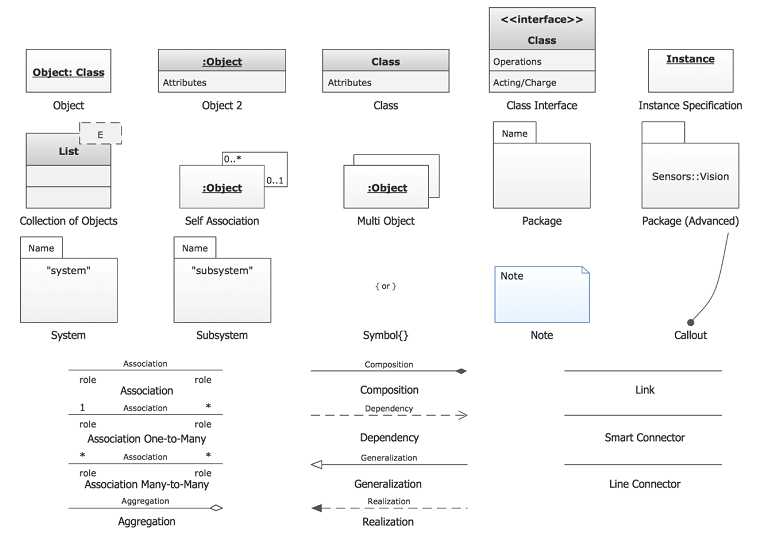
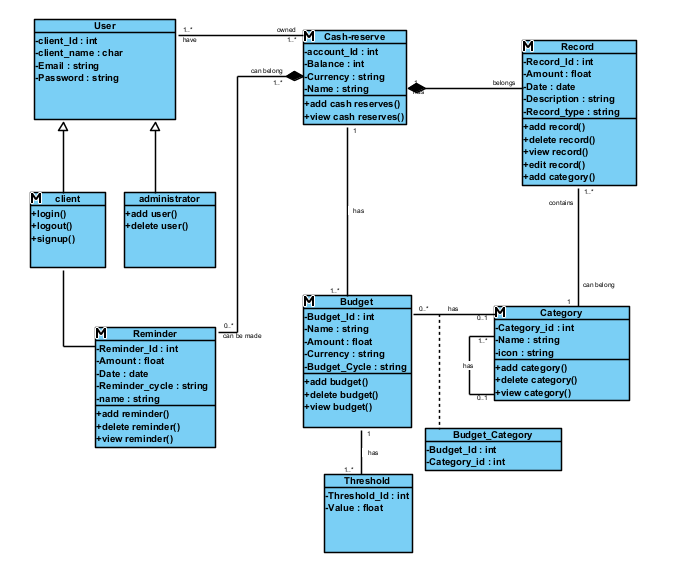
Components of a class diagram

Figure 32: Components of a class diagram

Figure 29:components of a class diagram

FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF



* + - * 1. Object diagram

An object diagram in the Unified Modelling Language (UML) is a diagram that shows a complete or partial view of the structure of a modeled system at a specific time. Object diagrams and class

diagrams are closely related and they use almost the same notation. Class diagrams model the rules for entities whereas the object diagrams model facts about specific entities.

Formalism of an object diagram

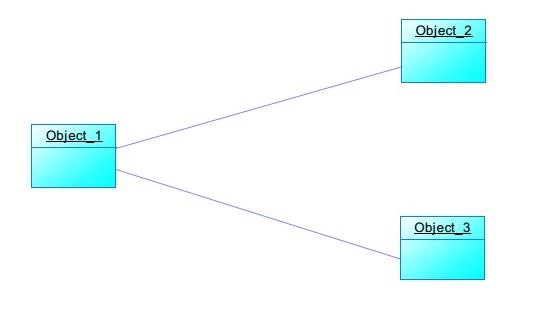


Figure 30:: Formalism of an object diagram

Components of an object diagram

Table 21: Components of object diagram

|  |  |  |
| --- | --- | --- |
| **Name** | **Description** | **Notation** |
| **Object name** | Every object is symbolized like a rectangle, that offers the name from the object and its class underlined as well as divided with a colon. |  |
| **Object attribute** | Similar to classes, you can list object attributes inside a separate compartment. However, unlike classes, object attributes should have values assigned to them. |  |
|  |  |  |
| **links** | Links tend to be instances associated with associations. You can draw a link while using the lines utilized in class diagrams. |  |

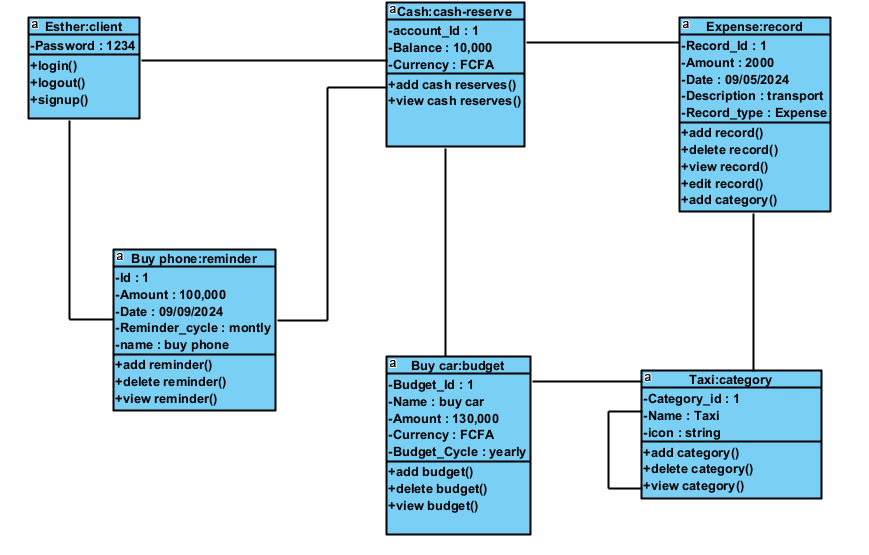


Figure 31:object diagram

# 

# PART FIVE: REALIZATION PHASE

**Preamble**

The realization phase contains the facts that are related to the analysis phase and aim at the physical implementation of the software.

INTRODUCTION

1. CHOICE FOR TECHNOLOGY PUT IN PLACE
2. ARCHITECTURE OF THE APLLICATION

CONCLUSION

## INTRODUCTION

The realization document contains the development of the most important features of the application. It completes the preceding document and any mistake made here will certainly alter the final application and might even undermine the work of the previous documents. Therefore, it will be a question here to present the physical data model of the application, the tools used for the realization of the application, the programming language chosen as well as the architecture of the application.

##### 

## CHOICE OF TECHNOLOGY PUT IN PLACE

### SOFTWARE RESOURCES

Table 22: Software resources

|  |  |  |  |
| --- | --- | --- | --- |
| **Tools Version** | | **Editor Description** | |
| Mongo DB | 3.2.4 | Apache Friends | Data base server |
| Gantt Project | 2.8.11 | Free Software  Foundation | Used for planning project planning |
| Visual Paradigm | 15.1 | Sybase Power AMC | Modelling software |
| Microsoft Office Word | 2019 | Open source  community | For typing/editing the report |
| V.S Code | 3 | V.S Code | Code editor and execution environment |
| Mozilla Firefox | 105.0 | Firefox | Web browser permitting interaction with the database server. |

### MATERIAL RESOURCES

* Windows 10 professional
* 4GB Ram memory
* 500GB hard drive
* Intel core 2 Duo processor
* 2.66Hz processor speed

## ARCHITECTURE OF THE APPLICATION

### PHYSICAL ARCHITECTURE

A 3-tier application architecture is a modular [client-server](https://www.techtarget.com/searchnetworking/definition/client-server) architecture that consists of a presentation tier, an application tier and a data tier. The data tier stores information, the application tier handles logic and the presentation tier is a graphical user interface [(GUI)](https://www.techtarget.com/whatis/definition/GUI) that communicates with the other two tiers. The three tiers are logical, not physical, and may or may not run on the same physical server.

**Presentation tier**: This tier, which is built with [HTML5,](https://www.techtarget.com/whatis/definition/HTML5) cascading style sheets [(CSS)](https://www.theserverside.com/definition/cascading-style-sheet-CSS) and [JavaScript,](https://www.theserverside.com/definition/JavaScript) is deployed to a computing device through a web browser or a web-based application. The presentation tier communicates with the other tiers through application program interface ([API)](https://www.techtarget.com/searchapparchitecture/definition/application-program-interface-API) calls.

**Application tier**: The application tier, which may also be referred to as the logic tier, is written in a programming language such as [Java](https://www.theserverside.com/definition/JavaScript) and contains the [business logic](https://www.techtarget.com/whatis/definition/business-logic) that supports the application's core functions. The underlying application tier can either be

hosted on distributed servers in the cloud or on a dedicated in-house server, depending on how much processing power the application requires.

**Data tier**: The data tier consists of a database and a program for managing read and write access to a database. This tier may also be referred to as [the storage tier](https://www.techtarget.com/searchstorage/definition/tiered-storage) and can be hosted

on-premises or in the cloud. Popular database systems for managing read/write access include [MySQL,](https://www.techtarget.com/searchoracle/definition/MySQL) PostgreSQL, Microsoft SQL Server and [MongoDB](https://www.techtarget.com/searchdatamanagement/definition/MongoDB).

### LOGICAL ARCHITECTURE

Taking into consideration that the platform has to be maintainable, the model we have chosen for the logical architecture of our system is the MVC (Model View Controller). It is a software design pattern for developing web applications and web pages.

In this part, we had as essential objective to show how our platform is going to be deployed, the use of tools such as bootstrap as framework of CSS (Cascading Style Sheet), XAMPP as web server and others that enabled us to implement the project. Now that our solution has been put in place, we can produce an installation guide and a user guide.

* **Model:** The lowest level of the pattern which is responsible for maintaining data.
* **View:** This is responsible for displaying all or a portion of the data to the user.
* **Controller:** Software code that controls all the interactions between the model and the view.

The view and the controller together comprise the user’s interface. The user interacts with the system solely through the view. Thus, the modification of the design, the presentation of the views will not affect the code of the models in relation to the database. In the same way, the modification of the models and the controllers will not affect the view. The diagram below illustrates the notion of MVC model.

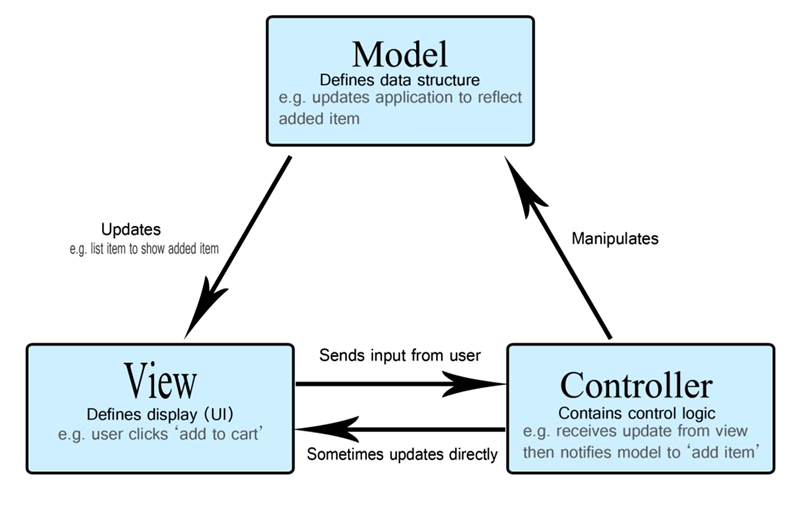
****

Figure 32:Model view controller (source: https://developer.mozilla.org/en-US/docs/Glossary/MVC )

## CONCLUSION

In this part we had as essential objective to show how our platform is going to be deployed, the use of tools such as Mongo DB as web server and others that enabled us to implement the project. Now that our solution has been put into place, we can produce an installation guide and a user guide.

# PART SIX: USER’S GUIDE

**Preamble**

Once a solution is analyzed, conceived, and deployed, it is now necessary to produce a guide for its users. This part of our document concerns the installation of the necessary tools to set up an environment for the platform in local and the user guide.

PLAN

INTRODUCTION

1. INSTALLATION
2. FUNCTIONALITY TEST
3. USERGUIDE

CONCLUSION

## INTRODUCTION

The user manual is the document created after the implementation of a software or platform or application. It answers the question “how to use the software application or platform that is in front of us?”. It is therefore crucial for us to offer the different users of our platform a guide allowing them to easily perform various operations on our platform.

## INSTALLATION GUIDE

The installation of the K-wallet application is not on play Store due to lack of finances, so the author will have to share the application through Xender.

## USER GUIDE

The user of the application will thus be able to carry out the following actions:

* Open the application:

Here, the user must tap on the logo present on the home page of their phone or menu

* After entering the application, the user will have two possibilities that is to Login or Sign Up on the platform.
* In case the you click on “Register Here”, he will have access to the registration platform where he has to fill his information and click on the “Register” button
* After the user is registered, he is being taken to the Login Page to authenticate
* After authenticating the user is taken directly to the dashboard

## CONCLUSION

In conclusion, this guide has provided us with the essential steps to effectively use the K-wallet application. By following the instructions and tips, you can maximize its features and improve your overall experience. Remember to refer back to this guide if you encounter any challenges. Enjoy the benefits of streamlined and efficient use!.

# GENERAL CONCLUSION

At the end of the Higher Technician Diploma (HTD), this internship enabled us not only to put into practice the knowledge acquired during our two years of training in computer sciences, but also specially to discover the professional world as well as the difficulties that we may face in our field in the nearest future. This internship was very enriching to the extent that it allowed us not only to discover in a general way problems that some companies of our society face, but also specially to prepare ourselves to avoid that our future companies should be confronted with the same problems. It is thanks to this very constructive internship experience that we have carried out in the enterprise, that enabled us to answer the question we had about how to achieve financial discipline in order to attain our goals and objectives. It should be noted that updates and important improvements will be made in the nearest future. This will not only improve our platform in terms of functionality, but also in security terms allowing an even more reliable platform

# GLOSSARY:

**1. Expense**

* **Definition**: Money spent on goods, services, or bills. This can include purchases, utility payments, and any other outflows of cash.

**2. Income**

* **Definition**: Money received, typically from a salary, business, or other sources like interest or investments.

**3. Budget**

* **Definition**: A financial plan that estimates income and expenses over a specified period, helping users manage their money and avoid overspending.

**4. Category**

* **Definition**: A classification used to group expenses and income. Examples include "Food," "Entertainment," "Housing," and "Salary."

**5. Transaction**

* **Definition**: A record of any financial activity, either spending (expense) or receiving (income), documented in the app.

**6. Balance**

* **Definition**: The remaining amount of money after accounting for all expenses and income. This can refer to a user’s total available funds.

**6. Cash Reserve**

* **Definition**: Money readily available for use

**6. Threshold**

* **Definition**: It can represent a boundary level, such as the minimum or maximum value needed for an event to occur or for a process to start.

# ANNEX:

# BIBLIOGRAPHY

* Mercuriale 2024, 1499 pages
* Learn Node js in 1 day by Krsihna Rungta, 105 pages
* Mongodb\_quick\_guide, 18 pages
* React js tutorial, 221 pages
* UML @ Classroom, 215 pages

# WEBOGRAPHY

* <https://reactjs.org/> : Provides a modern and efficient way to build dynamic user interfaces for web applications.
* <https://www.w3schools.com/> Easy-to-understand tutorials on various web development topics.
* <https://dribbble.com/> : for a clean and user-friendly UI/UX
* <https://www.freepik.com/search?format=search&last_filter=type&last_value=vector&query=girl%20sitting%20laptop&type=vector> :link to images for my design
* <https://rainerregan.medium.com/how-to-create-a-simple-pie-chart-with-chart-js-on-react-1e5a16f364a3> : To create a pie chart using react