**CHAPTER THREE**

**SYSTEM ANALYSIS AND DESIGN**

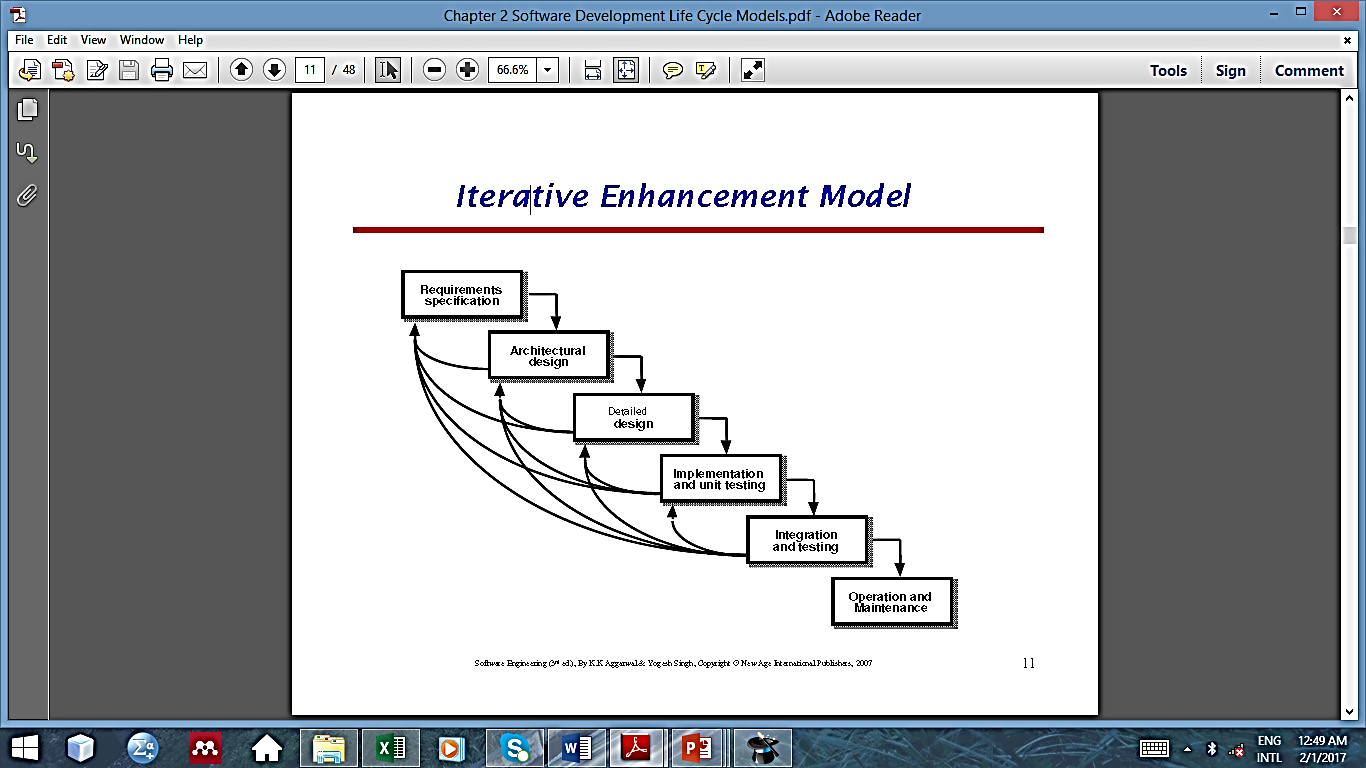
**3.1. INTRODUCTION**

Chapter Three is concerned with the research methodology employed in carrying out the research and analysis and design of the newly proposed system (electronic notice board).

**3.2 SOFTWARE DEVELOPMENT LIFE CYCLE (ITERATIVE)**

SDLC is the period of time that starts when a software product is conceived and ends when the product is no longer available for use (Aggarwal & Singh, 2007).

Iterative-Enhancement model shall be used for the purpose of this software developmental research. This is because the model provides the flexibility of moving to-and-fro any development stages of the software.

  
 **FIG 3.1:** Iterative-Enhancement model

**3.2.1 REQUIREMENT ELICITATION**

Requirement elicitation is the process of collecting the requirement of a system or requirement gathering information about a system from users, customers and stakeholders by conducting interview, questionnaire and brainstorming session (Aggarwal & Singh, 2007).

In this project, observation are the requirement elicitation techniques used in identifying the drawback of the manual methods of placing notice on pigeonholes.

* 1. **SYSTEM ANALYSIS**

System analysis or study is an important phase of any system development process. The system is viewed as a whole, the inputs are identified and the system is subjected to close study to identify the problem areas. The solutions are given as a proposal. The proposal is reviewed on user request and suitable changes are made. This loop ends as soon as the user is satisfied with the proposal.

**3.3.1 DESCRIPTION OF THE PROPOSED SYSTEM**

The proposed system intends to solve all the deficiency noted in the manual system. in order to improve information dissemination in department of computer science kaduna polytechnic, The Electronic Notice Board (E-Notioce Board) System will be design to engaged in providing up-to-date information to all the registered staffs across the department whenever there is new information, and also notify them via Short Message Service(SMS) through their handheld devices (phones or tablets). The Proposed System will be time saving, less anxiety as information can be known quickly, Records of information dissemination are kept for reference purposes and also the system will support feedback from staff in any case.

## **REQUIREMENT ANALYSIS**

Requirement analysis is the process of determining user expectations for a new or modified product. These features called requirement must be quantifiable, relevant and detailed. In software engineering such requirement are often called functional specifications. Requirement analysis is an important aspect of project management.

Requirement engineering involves frequent communication with system users to determine specific feature expectations, resolution of conflict or ambiguity in requirements as demanded by various users or group of users, avoidance of feature creep and documentation of all aspects of the project development process from start to finish. More attention should be directed towards ensuring the final system or product conforms to user’s needs rather than attempting to make user’s expectations to fit the requirements.

**3.4.1 FUNCTIONAL REQUIREMENT**: Functional requirements in software engineering define a function of a system or its component. Functional requirements are those requirements which deal with what the system should do or provide for users.

The functional requirements for e-notice board system are as follows:

1. The system should allow admin to register new staff.
2. The system should allow staffs and admin to login and logout.
3. The system should allow admin to compose notice to registered staffs.
4. The system should allow staffs to view notice sent from admin.
5. The system should allow the staffs to send enquiry to admin.
6. The system should allow the admin to respond to enquiry.
7. The system should allow admin and staff to update their profile.
8. The system should allow admin to edit and delete staff.
9. The system should notify staffs via short massage service(sms)when new notice is sent to them from admin.

**3.4.2NON-FUNCTIONAL REQUIREMENT**: These kinds of requirements are sometimes called constraints of the system. Non-functional requirements denote limits of the system and its expected behavior. They do not impact the system directly in terms of functionality, the following are the Non-functional requirement of the notice/letters dissemination system:

I. Availability: The system should be available to the users at all time.

II. Security: The system should be secured to avoid unwanted access.

III. Reliability: The system should be reliable in such a way that it performs its tasks properly at all time without producing any ambiguous result.

IV. Scalability: The system should be able to handle the task as number of users increase.

V. Timeliness: This requirement ensures continuous available and quicker and always prompt in response for every time it’s in use.

**3.4.1 USE CASE**

A use case is a methodology used to identify, clarify, and organize system requirements. The use case is made up of a set of possible sequences of interactions between systems and users in a particular environment and related to a particular goal. It consists of a group of elements (for example, classes and interfaces) that can be used together in a way that will have an effect larger than the sum of the separate elements combined. The use case should contain all system activities that have significance to users.

Electronic Notice Board

admin

staff

**fig 3.2: Use Case Diagram.**

**3.5.2 Use Case Description**

Use case description is a detailed description of each Use Case; listing the actor(s) that interact with the use case, a brief description about the current Use Case, conditions that must be met before and after a Use Case can be and has been accessed, as well as the basic and alternate flow of the system depending on the interaction.

**Table 3.1: Use Case Description**

|  |  |
| --- | --- |
| **Use Case:** | Log In |
| **Actor:** | Staff/Admin |
| **Introduction:** | This section describes the procedure of logging into the system |
| **Pre-Condition:** | None |
| **Post-Condition:** | If the operation is successful, user is logged into the system otherwise, user receives error message. |
| **Basic Flow:** | 1. User is prompted to enter username and password 2. User enters username and password 3. System validates both username and password and finds them to be in correct form and valid 4. User is logged into the system 5. Use Case ends |
| **Alternate Flow:** | 1. User is prompted to enter username and password 2. User enters username and password 3. System validates both username and password and finds them either not in correct format or invalid 4. Error message is returned to the user 5. System state remains unchanged 6. Use Case ends |

|  |  |
| --- | --- |
| **Use Case:** | Compose Notice |
| **Actor:** | Admin |
| **Introduction:** | This section describes the procedure of disseminating information |
| **Pre-Condition:** | Admin must log in to the system |
| **Post-Condition:** | If the operation is successful, notice will be broadcast to intended staff otherwise, the system will remain unchanged. |
| **Basic Flow:** | 1. Admin navigate to compose form. 2. Admin select/input the recipient username from staffs list. 3. System validate the admin inputs and the recipient username. 4. System will display message if the notice is send. 5. Use Case ends. |
| **Alternate Flow:** | 1. Admin navigate to compose form. 2. Admin select/input the recipient username from staffs list. 3. System validate the admin inputs and the recipient username. 4. System validates the username and admin inputs, it finds out that either the username not registered or wrong format input. 5. Error message is returned to the admin. 6. System state remain unchanged. 7. Use case ends. |
| **Use Case:** | Register new staff. |
| **Actor:** | Admin. |
| **introduction** | Admin navigate through the staff registration form to add new staff. |
| **Pre-condition:** | Admin must log in to the system |
| **Post-condition:** | If the registration is successful, new staff will be added otherwise, the system will remain unchanged. |
| **Basic flow:** | 1. Admin navigate to staff registration page. 2. Admin inputs all the necessary information of the staff. 3. System validate the inputs to make sure that the staff is not exit in the system. 4. Admin click on registered button to add new staff. 5. The system displays success message to the admin. 6. Use case ends. |
| **Alternate flow:** | 1. Admin navigate to staff registration page. 2. Admin inputs all the necessary information of the staff. 3. System validates the inputs to make sure that the staff is not exit in the system, if staff already exit. 4. Error message is returned to the admin. 5. System state remains unchanged. 6. Use case ends. |
| **Use case:** | Delete staff |
| **Actor:** | Admin |
| **Introduction:** | Admin navigate to staff reports page to delete and select the staff he wish to delete. |
| **Pre-condition:** | Admin must log in to the system |
| **Post-condition:** | If the operation is successful, staff’s record is deleted completely. Otherwise, system state remains unchanged. |
| **Basic flow:** | 1. Admin request to the system to delete a particular record. 2. System requests the admin to authenticates the operation by selecting **yes**, to go ahead with the operation, or **no**, from a prompt to abort the operation. 3. System removes the corresponding record from the system. 4. System returns a success message to the admin. 5. Use case ends. |
| **Alternate flow:** | 1. Admin request to the system to delete a particular record. 2. System requests the admin to authenticates the operation by selecting **yes**, to go ahead with the operation, or **no**, from a prompt to abort the operation. 3. User selects **no.** 4. System aborts the operation and returns a message to the user. 5. System state remains unchanged. 6. Use case ends. |
| **Use case :** | Update profile. |
| **Actor** | Admin/staff. |
| **Introduction** | This describe the step follow by the admin/staff to update his/her profile. |
| **Pre-condition:** | Admin/staff must log in to the system. |
| **Post-condition:** | If the operation is successful, the selected record will be uploaded and success message will return to the admin/staff. |
| **Basic flow:** | 1. Admin/staff navigate to update page. 2. Admin/staff make changes to his/her records. 3. System validate the inputs made by the admin/staff. 4. System check whether username already exit. 5. System update the database with the new information given. 6. System returns success message of the update 7. Use case ends. |
| **Alternate flow:** | 1. Admin/staff make changes to his/her records. 2. System validate the inputs made by the admin/staff. 3. System check whether username already exit, if the username exit or admin/staf provide wrong input to the system. 4. System displays error message 5. System remain unchanged. 6. Use case ends. |
| **Use case:** | View staff records |
| **Actors :** | Admin |
| **Introduction :** | This section will describe the process follow by admin to display the record of the registered staff. |
| **Pre-condition:** | Admin must log in to the system. |
| **Post condition:** | If the operation is successful, the system will display the record of registered staff. |
| **Basic Flow:** | 1. Admin navigate to staff report page 2. Admin click on view button to display a particular record. 3. Use case ends. |
| **Alternate flow:** | 1. Admin navigate to staff report page 2. Admin click on view button to display a particular record, if the record is not display then the record is not in the database. 3. System remains unchanged. 4. Use case ends. |
| **Use case:** | Create admin. |
| **Actor :** | Admin. |
| **Introduction :** | This section describe the operation follow by registered admin to create new admin. |
| **Pre-condition :** | Admin must log in to the system. |
| **Post-condition:** | If the operation is successful, new admin will be created otherwise an error message will pop up. |
| **Basic flow :** | 1. Admin navigate to create admin page. 2. Admin input the information of new admin. 3. System validate the admin input 4. System created new admin base on the valid information received. 5. System return success message. 6. Use case ends. |
| **Alternate flow :** | 1. Admin navigate to create admin page. 2. Admin input the information of new admin. 3. System validate the admin input and find out that the input is invalid. 4. System display error message. 5. System remains unchanged. 6. Use case ends. |
| **Use case :** | Make inquiry. |
| **Actor :** | Staff. |
| **Introduction :** | This section describe the process follow by staff to make inquiry to admin. |
| **Pre-condition :** | Staff must log in to the system |
| **Post-condition :** | If the operation is successful, the inquiry will be delivered to admin and staff will receive a success message. |
| **Basic flow :** | 1. Staff navigate to make inquiry page. 2. Staff select/input the admin username and make the inquiry. 3. The system validate the staff input. 4. System returns success message 5. Use case ends. |
| **Alternate flow:** | 1. Staff navigate to make inquiry page. 2. Staff select/input the admin username and make the inquiry. 3. The system validate the staff input, if the input is invalid. 4. System returns error message. 5. System remain unchanged. 6. Use case ends. |
| **Use case :** | View feedback. |
| **Actor :** | staff. |
| **Introduction :** | This process will describe the process follow by staff to view feedbacks. |
| **Pre-condition :** | staff must log in to the system. |
| **Post-condition :** | The system will display the feedback sent from admin. |
| **Basic flow :** | 1. Staff navigate to the view feedback page. 2. Staff click on the feedback message from the list of messages. 3. System display the feedback message 4. Use case ends. |
| **Alternate flow :** | 1. Staff navigate to the view feedback page. 2. Staff click on the feedback message from the list of messages, if there is no feedback the system display null feedback. 3. System remains unchanged. 4. Use case ends. |
| **Use case :** | Log out. |
| **Actor :** | Admin/Staff. |
| **Introduction :** | This section describes the procedure of logging out of the system. |
| **Pre-condition :** | User must be logged-in to the system. |
| **Post-condition :** | If the operation is successful, user is logged out of the system otherwise, system state remains unchanged. |
| **Basic flow :** | 1. User clicks on the Logout button. 2. User is prompted to confirm logout request. 3. User chooses **yes.** 4. User is logged out of the system. 5. Use Case ends. |
| **Alternate flow :** | 1. User clicks on the Logout button. 2. User is prompted to confirm logout request. 3. User chooses **cancel.** 4. System state remains unchanged. 5. Use Case ends. |

**3.5 SYSTEM DESIGN**

System design is the process of defining the components, modules, interfaces, and data for a system to satisfy specified requirements. The design of this system would be shown using a class diagram and an activity diagram.

The system design can be captured from different perspectives as well. There are many concepts providing guidelines to design and model system. For example, UML allows capturing system design by package.

* + 1. **Class Diagram**

A class diagram in the unified modelling language (UML) is a type of static structure diagram that describes the structure of a system by showing the systems classes, their attributes, operations (or methods), and the relationship among objects. The class diagram for the E-notice board System is given below.

Admin

-admnId:int

-adminName:varchar

-Email:varchar

-adminUsername:varchar

-adminPassword:varchar

-adminPhonenu:varchar

+composeNotice()

+registerStaff()

+registerAdmin()

+viewStaffrecord()

+editStaffrecord()

+deleteStaffRecord()

+updateProfile()

+viewInquiry()

1

**Fig 3.3 Class Diagram**

\*

viewNotice

-noticeId:int

-Notice:varchar

-date:date and time

-senderUsername:varchar

+staff()

Staff

-staffId:int

-staffName:varchar

-staffEmail:varchar

-staffUsername:varchar

-staffPassword:varchar

-staffPhonenumber:varchar

-staffDob:date and time

-staffLevel:varchar

+viewNotice()

+makeInquiry()

+viewfeedback()

+updateProfile()

Login

-userName:varchar

-password:varchar

+login()

+logout()

Logout

+admin()

+staff()

Registerstaff

+register()

**3.5.4 ACTIVITY DIAGRAM**

An activity diagram in the unified modelling language (UML) used to describe the dynamic aspect of a system. Activity diagram is basically a flow chart to represent the flow from one activity to another activity. The activity can be described as an operation of the system.

**Admin E-notice Board System**

Login

Validate

Dashboard

Invalid input

Valid input

Compose Notice

View Feedback

View Staff Reports

Logout

Update Profile

Register Staff

Create Admin

Delete Staff

**Fig 3.4 Activity Diagram for Admin**

**Staff E-notice board system**

Login

Validate

Dashboard

Invalid input

Valid input

View Notice

Make inquiry

Update Profile

Logout

**Fig 3.4 Activity Diagram for Staff**