EXPERIMENTAL STUDY OF ELECTRONIC KEY MOVEMENT BOOK FOR ORGANIZATIONS

Japheth B. R. and Patrick K.
Department of Mathematics/Computer Science
Niger Delta University, Yenagoa, Nigeria
jbunakiye@yahoo.com; ktarila@gmail.com

ABSTRACT

Usually collection of keys are manually driven, hence it's time consuming. Staffs of organizations have to sign in and sign out for keys in order to accomplish the daily jobs. In the process much time is spent collecting keys for the specific offices. This is because they are expected to write their names, departments, date, time against these respected columns; before they will now sign across the key columns, in a Key Movement Book (KMB) that keeps records keys collection. The present system is a manual driven process, where signing in and out of key is done manually. Hence, the signing of keys is slow and in efficient in terms of checking for keys availability and retrieval. In addition, the naming convention of the keys are don randomly and not in a structured manner. The collection of keys is insecure. This means the security risk of a staff coming from one department to sign key for another department is high. This risk coupled with the associated job hazards of the security staff necessitated the analysis of this new with a view to minimizing or out rightly stopping them. The requirements and structure of the new system is analyzed and designed. The requirement entails component parts of the systems. While the structure describes the system design, in respect to Entity Relational Diagram(ERD), Data Flow Diagram (DFD), Class Diagram(CD). These diagrammatic tools are used to show the, inputs, objects, processes and outputs of systems in a diagrammatic manner.

Key Words: Entity Relational Diagram (ERD), Data Flow Diagram (DFD), Key Movement Book (KMB), Class Diagram (CD), Key Movement Report (KMR), Key Report Book (KRB)

INTRODUCTION

The present system is a manual driven process, where signing in and out of key is done manually. Hence, the signing of keys is slow and in efficient in terms of checking for keys availability and retrieval. The manual generation of this Key Movement Report (KMR) [11] is tedious, because the security staff has to open many pages of the Key Report Book (KRB) just to get an entire record for a department; hence the procedure is not appropriate in information gathering; because it is highly time consuming for a security officer; and he/she can't afford to do it. The generation of key movement report history [11, 10] is an important fact in organizational security system. The security department will use the key movement history to keep track of staff and keys for a particular day and time. This report will help the security department in staging a good investigation when the need arises. The ERD is a relational diagram which defines the schematic relationships between database tables. The relationships between the tables are explained in the diagram, symbolically. It's actually a

part of the described data collected when gathering the information. The DFD is an information flow representation diagram, showing the flow of data from source to destination, and destination to source. It specifies the availability of data to a particular component in the system. This helps the designer to know what a component is expected to do and how to do it. The CD is an object diagram that represents the objects used in the systems. It shows the properties and methods (functions) this objects are made up of [3].

REQUIREMENT ANALYSIS

The system requirements in this new system are input requirements and the output requirements [4, 3]. The input requirements are the component parts that are needed to build the new system, the information that is needed to be stored or processed in the database and are broken down follows:

- Staff personal information
- Staffs contact information
- Staff departmental details
- Security staff details
- Department and offices details
- Offices, stores and lab details
- Key information details
- Security Staff Comments

The system is required to output the following

- Key Report History
- Viewing of Users information before key collection
- Viewing of Signed and Available Keys
- Viewing of System Users Information
- Viewing Department Details
- Viewing all Offices, Labs and Store keys information
- Viewing Security Staff closing comments of the day

SYSTEM DATABASE REQUIREMENTS

The new system uses a database server, Microsoft SQL Server 2005, Expression Edition to store all the required information. Unlike the present systems where information are stored in a text book, the database server stores information and also allow easy access and retrieval of information when need arises. The generation of reports from the database is easy and efficient, because the information stored can easily be accessed by querying the database with a language called Structured Query Language (SQL). A relationship is some association between entities. That is, associations between two or more entities [5, 6, and 3]. An entity is an abstraction from the complexities of some domain. When we speak of an entity we normally speak of some aspect of the real world which can be distinguished from other aspects of the real world. The association is done using database table diagrams (entities

diagram) and the relationship is shown by using their foreign to primary key fields. The following are the tables used in designing the system [1, 2].

Staff Personal Information Table: The Staff Personal Information table is used to store staff information necessary for the security department in terms of key collection and returning.

Table 3.1 StaffPersonalInformation *Table*

Autoincrement	Primary Key	Column Name	Data Type	Allow Null
Yes	Yes	StaffID	int	Not Null
		StaffNumber	nvarchar(50)	Not Null
		Title	nvarchar(10)	Not Null
		Surname	nvarchar(50)	Not Null
		FirstName	nvarchar(50)	Not Null
		Othernames	nvarchar(50)	Null
		Sex	char(1)	Not Null
		DepartmentUnitId	int	Not Null
		Passport	image	Not Null

StaffId: The StaffId field provides the unique identifying number for a specific the number of staff in the organization. This field is the primary key and has an auto increment property; in addition, as a result of being a numeric value, its data type is an int, and the field will not allow any null values.

StaffNumber: The StaffNumber is a unique key field(meaning no two employee will have the same staff number) , it is use for storing of the staff number (a number that will be assign to each staff of the organization); in addition, its of type nvarchar(50) and it doesn't allow null values.

Title: The Title field stores the value of the staff title(which are Mr., Miss., Dr, Pro, Chief etc.); in addition, it's of type nchar(10) and it doesn't allow null values.

Surname: The Surname field stores the value of the staff surname; in addition, it's of type nvarchar(50) and it doesn't allow null values.

FirstName: The FirstName field stores the value of the first name of the staff; in addition, it's of type nvarchar(50) and it doesn't allow null values.

Othernames: The Othernames field stores the value of the other names of the staff; in addition, its of type nvarchar(50) and it allows null values, because not everyone has another name apart from their first name and last name.

Sex: The Sex field stores the value of the staff sex; in addition, it's of type char (1) and it doesn't allow null values.

DepartmentUnitId: The DepartmentUnitId is a foreign key to the DepartmentUnit table. This means any value entered here is check against the value that exist in the DepartmentUnit table. In addition, it's of type int (because it store only integer values) and it doesn't allow null values.

Passport: The Passport field stores the value of the staff passport image; in addition, it's of type image (it is seen as byte) and it doesn't allow null values.

DailyEvent Table: This table is use for storing information of security staffs events of the day. That is the activities carried out in respect to the key collection and returning of university staff; when they login and out of the application daily.

Table 3.2 ***DailyEvent Table***

AutoIncrement	Primary Key	Column Name	Data Type	Allow Null
Yes	Yes	DaillyEventId	int	Not Null
		SecurityStaffId	int	Not Null
		LogOnTime	datetime	Not Null
		LogOutTime	datetime	Not Null
		ClossingReport	text	Not Null
		ClosingRemark	char(1)	Not Null

DaillyEventId: The DailyEventId field provides the unique identifying number for a specific the number of daily remarks by each security staff before closing for each day. This field is the primary key and has an auto increment property; in addition, as a result of being a numeric value, its data type is an int, and the field will not allow any null values.

SecurityStaffId: The SecurityStaffId is a foreign key to the StaffPersonalInformation table. This means any value entered here is check against the value that exist in the StaffPersonalInformation table. In addition, it's of type int (because it store only integer values) and it doesn't allow null values.

LogOnTime: The LogOnTime field stores the value of the date and time the security staff signs into the application; in addition, its of type datetime and it doesn't allow null values.

LogOutTime: The LogOnTime field stores the value of the date and time the security staff signs out of the application; in addition, it's of type datetime and it doesn't allow null values.
ClossingReport: The ClossingReport field stores the value of the staff closing security report in respect to what has happen through the period when he/she is on duty; in addition, it's of type "text" and it doesn't allow null values.

ClosingRemark: The ClosingRemark field stores the value of the closing remark checker (that is, if report was entered or not before signing out) that accepts 'y' or 'n' as input. When staff login it will contain 'n' and when staff writes a closing remark and submit, the value changes to 'y'; in addition, it's of type char(1) and it doesn't allow null values.

UserDetails Table: This table is use for storing information of security staffs that will have access to the application. It contains the staff log on information.

Table 3.3 ***UserDetails Table***

Autoincrement	**Primary Key**	**Column Name**	**Data Type**	**Allow Null**
Yes	Yes	UserDetailsId	int	Not Null
	Unique Key	StaffID	int	Not Null
	Unique Key	UserName	nvarchar(50)	Not Null
	Case Sensitive	Password	nvarchar(50)	Not Null
		Authority	char(20)	Not Null

UserDetailsId: The UserDetails field provides the unique identifying number for a specific the number of security users who has access to the application. This field is the primary key and has an auto increment property; in addition, as a result of being a numeric value, its data type is an int, and the field will not allow any null values.

StaffID: The SecurityStaffId is a foreign key to the StaffPersonalInformation table and its is also a unique field (that is no two the same security staff ID can be entered in this field. This means any value entered here is check against the value that exist in the StaffPersonalInformation table. In addition, it's of type int (because it store only integer values) and it doesn't allow null values.

UserName: The UserName is a unique key field(meaning no two employee will have the same username) , it is use for storing of the username (a unique name that will be assign to

each security staff of the university, that will make use of that application); in addition, it's of type nvarchar(50) and it doesn't allow null values.

Password: The Password is a case sensitive field(which means Mic is no equal to mic nor miC, since the case of the characters are not in the same formart) , it is use for storing of passwords of security staff; in addition, it's of type nvarchar(50) and it doesn't allow null values.

Authority: The Authority field stores the value of the staff administrative authorization (which is Administrator or User) of the application. That is, only administrator are given full access right(rights to add, modified and even delete information from the database) to the application; in addition, it's of type nchar(20) and it doesn't allow null values.

RoomDetails Table: This table is uses for storing information of all the rooms in the university.

Table 3.4 ***RoomDetails Table***

AutoIncrement	**Primary Key**	**Column Name**	**Data Type**	**Allow Null**
Yes	Yes	RoomId	int	Not Null
		RoomType	nchar(20)	Not Null
		RoomName	nchar(30)	Not Null
	Unique Key	KeyBondleCode	nchar(10)	Not Null
		DepartmentUnitId	int	Not Null

RoomId: The RoomId field provides the unique identifying number for a specific the number of rooms in the university. This field is the primary key and has an auto increment property; in addition, as a result of being a numeric value, its data type is an int, and the field will not allow any null values.

RoomType: The RoomType field stores the value of the room type(which are, Offices, Lab, Stores etc); in addition, its of data type nchar(20) and it doesn't allow null values.

RoomName: The RoomName field stores the value of the room name (which are Computer Lab 1, Physics Lab 1, Chemistry Lab 1, Establishment Store etc.); in addition, its of type nchar(30) and it doesn't allow null values.

KeyBondleCode: The KeyBondleCode is a unique key field(meaning no two keybondle will have the same keybondle code) , it is use for storing of the key code (a code that will be assign to each key bundle that is assign to a room); in addition, its of type nchar(10) and it doesn't allow null values.

DepartmentUnitId: The DepartmentUnitId is a foreign key to the DepartmentUnit table. This means any value entered here is check against the value that exist in the DepartmentUnit table. In addition, its of type int (because it store only integer values) and it doesn't allow null values.

KeyDescription Table: This table is use for storing information of all the keys in the university. It also contains a field that will store the room ids of each key in that particular room type.

Table 3.5 ***KeyDescription Table***

AutoIncrement	**Primary Key**	**Column Name**	**Data Type**	**Allow Null**
Yes	Yes	KeyId	int	Not Null
	Unique Key	KeyCode	nchar(10)	Not Null
		NumbersOfKeys	int	Not Null
		RoomId	int	Not Null

KeyId: The KeyId field provides the unique identifying number for a specific the number of keys locks in the university. This field is the primary key and has an auto increment property; in addition, as a result of being a numeric value, its data type is an int, and the field will not allow any null values [7, 8, and 9].

KeyCode: The KeyCode is a unique keycode field(meaning no two key will have the same key code) , it is use for storing of the key code (a code that will be assign to each group of key of a particular lock of a room); in addition, its of type nchar(10) and it doesn't allow null values.

NumbersOfKeys: The NumberOfKeys field stores the value of the quantity of keys of a particular lock of a room; in addition, its of type datetime and it doesn't allow null values.

RoomId: The RoomId is a foreign key to the RoomDetails table. This means any value entered here is check against the value that exist in the RoomDetails table. In addition, it's of type int (because it store only integer values) and it doesn't allow null values.

KMCollectionHistory Table: This table is use for storing information of all keys collection and returning transactions. It also contains the entire required field for key collection and returning as specified in the manual process.

Table 3.6 ***KMCollectionHistory Table***

AutoIncrement	Primary Key	Column Name	Data Type	Allow Null
Yes	Yes	KMHistoryID	int	Not Null
		RoomId	int	Not Null
		WhoCollectID	int	Not Null
		DateCollected	datetime	Not Null
		TimeCollected	datetime	Not Null
		WhoReturnID	int	Null
		DateReturned	datetime	Null
		TimeReturned	datetime	Null
		Remark	bit	Not Null

KMHistoryID: The KMHistory field provides the unique identifying number for a specific the number of keys transaction that has been carried out. This field is the primary key and has an auto increment property; in addition, as a result of being a numeric value, its data type is an int, and the field will not allow any null values.

RoomId: The RoomId is a foreign key to the RoomDetails table. This means any value entered here is check against the value that exist in the RoomDetails table. In addition, it's of type int (because it store only integer values) and it doesn't allow null values.

WhoCollectID: The WhoCollectID is a foreign key to the StaffPersonalInformation table. This means any value entered here is check against the value that exist in the StaffPersonalInformation table. In addition, it's of type int (because it store only integer values) and it doesn't allow null values.

DateCollected: The DateCollected field stores the value of the date the staff signs for key(s); in addition, it's of type datetime and it doesn't allow null values.

TimeCollected: The TimeCollected field stores the value of the time the staff signs for key(s); in addition, it's of type datetime and it doesn't allow null values.

WhoReturnID: The WhoReturnID is a foreign key to the StaffPersonalInformation table. This means any value entered here is check against the value that exist in the StaffPersonalInformation table. In addition, it's of type int (because it store only integer values) and it doesn't allow null values.

DateReturned: The DateCollected field stores the value of the date the staff signs in the key(s); in addition, it's of type datetime and it doesn't allow null values.

TimeReturned: The TimeReturned field stores the value of the time the staff signs in the key(s); in addition, it's of type datetime and it doesn't allow null values.

Remark: The Remark field stores the value of the state (meaning if the key is return or not) of the key; in addition, it's of type 'bit' (that is either 0 or 1, '0' means it hasn't been return while '1' means it has been return) and it doesn't allow null values.

DepartmentUnit Table: This table is use for storing information of departments units.

Table 3.7 ***DepartmentUnit Table***

AutoIncrement	Primary Key	Column Name	Data Type	Allow Null
Yes	Yes	DepartmentUnitId	int	Not Null
		DepartmentID	int	Not Null
		UnitName	nvarchar(50)	Not Null

DepartmentUnitId: The DepartmentUnitId field provides the unique identifying number for a specific the number of department units in the university. This field is the primary key and has an auto increment property; in addition, as a result of being a numeric value, its data type is an int, and the field will not allow any null values.

DepartmentID: The DepartmentID is a foreign key to the Department table. This means any value entered here is check against the value that exist in the Department table. In addition, it's of type int and it doesn't allow null values.

UnitName: The UnitName field stores the values of the units within each department; in addition, its of data type nvarchar (50) and it doesn't allow null values.

Department Table: This table is use for storing all departments' names of the university.

Table 3.8 ***Department Table***

AutoIncrement	Primary Key	Column Name	Data Type	Allow Null
Yes	Yes	DepartmentID	int	Not Null
		DepartmentName	nvarchar(50)	Not Null

DepartmentID: The DepartmentId field provides the unique identifying number for a specific the number of departments in the university. This field is the primary key and has an auto increment property; in addition, as a result of being a numeric value, its data type is an int, and the field will not allow any null values.

DepartmentName: The DeparmentName field stores the values of the names of each department within the university; in addition, its of data type nvarchar (50) and it doesn't allow null values.

THE RELATIONSHIPS

DailyEvent Table: The DailyEvent table has a single relationship, as described in table 4.0

Table 4.0 ***DailyEvent Relationships***

Constraint Name	Foreign Key	Reference Table	Reference Field
FK_DailyEvent_StaffPersonalInfo	SecurityStaffID	StaffPersonalInfo	StaffID

This relationship ensures that the SecurityStaffID field of the DailyEvent table will have a corresponding existing value within the StaffPersonalInformation table

UserDetails Table: The DailyEvent table has a single relationship, as described in table 4.1.

Table 4.1 ***UserDetails Relationships***

Constraint Name	Foreign Key	Reference Table	Reference Field
FK_UserDetails_StaffPersonalInfo	StaffId	StaffPersonalInfo	StaffId

KeyDescription Table: The KeyDescription table has a single relationship, as described in 4.2

Table 4.2 ***KeyDescription Relationships***

Constraint Name	Foreign Key	Reference Table	Reference Field
FK_KeyDescription_RoomDetails	RoomId	RoomDetails	RoomId

FK_ KeyDescription_RoomDetails: This relationship ensures that the RoomId field of the KeyDescription table will have a corresponding existing value within the RoomDetails table.

KMCollectionHistory Table: The **KMCollectionHistory** table has a three separate relationship, as described in table 4.3.

Table 4.3 *KMCollectionHistory Relationships*

Constraint Name	Foreign Key	Reference Table	Reference Field
FK_KMCollectionHistory_RoomDetails	RoomId	RoomDetails	RoomId
FK_KMCollectionHistory_StaffPersonalInformation	WhoCollectId	StaffPersonalInformation	StaffId
FK_KMCollectionHistory_StaffPersonalInformation1	WhoReturnId	StaffPersonalInformation	StaffId

FK_ KMCollectionHistory_RoomDetails: This relationship ensures that the RoomId field of the KMCollectionHistory will have a corresponding existing value within the RoomDetails table.

FK_ KMCollectionHistory_StaffPersonalInformation: This relationship ensures that the WhoCollectId field of the KMCollectionHistory will have a corresponding existing value within the StaffPersonalInformation table.
FK_ KMCollectionHistory_StaffPersonalInformation1: This relationship ensures that the WhoReturnId field of the KMCollectionHistory will have a corresponding existing value within the StaffPersonalInformation table.

StaffPersonalInformation Table: The **StaffPersonalInformation** table has a single relationship, as described in table 4.4.

Table 4.4 StaffPersonalInformation *Relationships*

Constraint Name	Foreign Key	Reference Table	Reference Field
FK_StaffPersonalInfo_DepartmentUnit	DepartmentUnitId	DepartmentUnit	DepartmentUnitId

FK_ StaffPersonalInformation_DepartmentUnit: This relationship ensures that the DepartmentUnitId field of the StaffPersonalInformation will have a corresponding existing value within the DepartmentUnit table.

RoomDetails Table: The **RoomDetails** table has a single relationship, as described in table 4.5.

Table 4.5 ***RoomDetails Relationships***

Constraint Name	Foreign Key	Reference Table	Reference Field
FK_RoomDetails_DepartmentUnit	DepartmentUnitId	DepartmentUnit	DepartmentUnitId

FK_UserDetails_StaffPersonalInformation: This relationship ensures that the DepartmentUnitId field of the RoomDetails will have a corresponding existing value within the DepartmentUnit table.

DepartmentUnit Table: The **DepartmentUnit** table has a three separate relationship, as described in table 4.6.

Table 4.6 ***DepartmentUnit Relationships***

Constraint Name	Foreign Key	Reference Table	Reference Field
FK_DepartmentUnit_Department	DepartmentId	Department	DepartmentId

FK_ DepartmentUnit _ Department: This relationship ensures that the DepartmentId field of the DepartmentUnit will have a corresponding existing value within the Department table.

Table Entity Relationship

This relationship as captured in figure 4.0 is normally at the conceptual view of the table. It shows how the various tables relate with one another since all the tables have information that can be directly linked to a department unit.

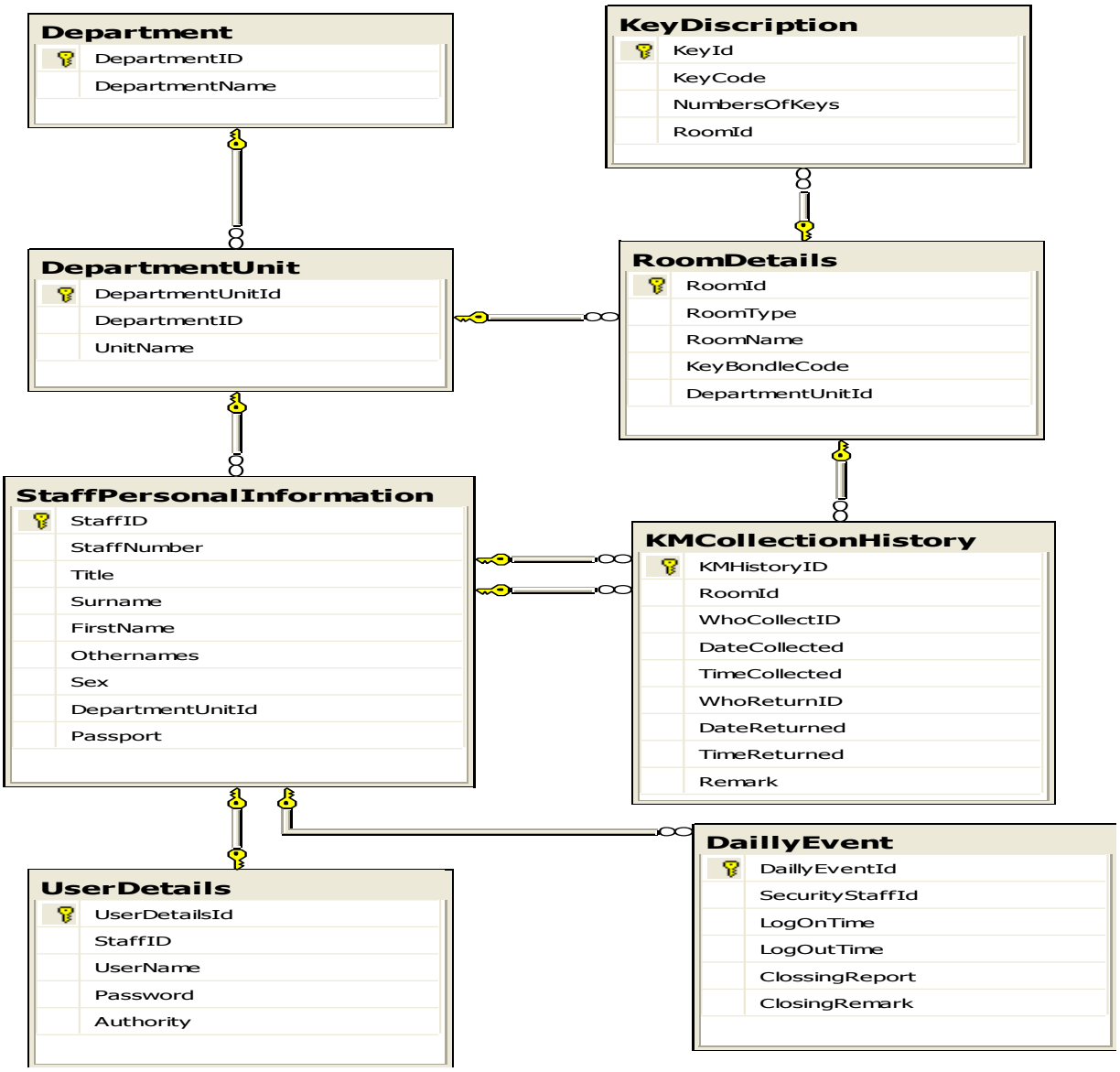

Fig. 4.0 Entity Relationship Diagram Of Key Movement System

DEPLOYMENT STEPS

The following are the system execution steps.

Step 1: Install the Program

Open the CD and click on the setup file to begin the installation process.

Step 2: Start the Program

Click Start -> Program -> Name of the Program (Key Movement System). Enter username and password on the logon window as shown in figure 5.0.

Fig 5.0 LogOn window

Step 3: Log In to the Main Application Window

If the username and password entered is valid then the user will be authorized to see the next window shown in figure 5.1.

Fig 5.1 KeyCollectionReturn window

The user can apply appropriately for automated service delivery on the application window.

Step 4: Staff Personal Information

Staff Personal Information windows contains any information that is relevant to the staff; that is needed by the security staff. Here the user can enter new staff record, edit, or update staff records as shown in figure 5.2.

Fig 5.2 StaffPersonalInformation window

Step 5: Room Details

Room details windows shown in figure 5.3 contains any information that is relevant to the organization pertaining to rooms, offices etc. that is needed by the security staff. Here the user can enter new rooms record, edit, or update room records in respect to the departments that these rooms are assigned to.

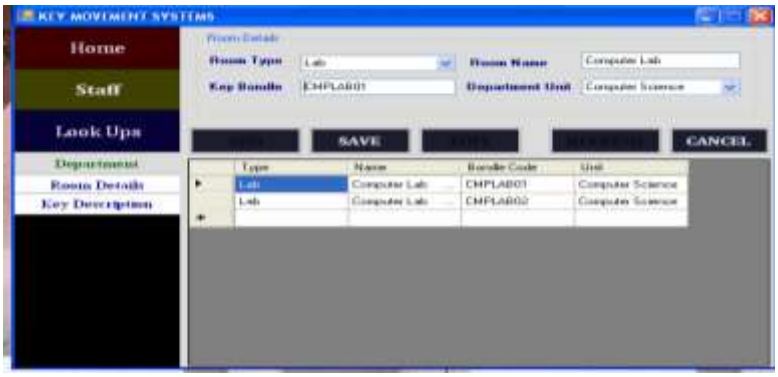

Fig 5.3 RoomDetails window

CONCLUSION

The generation of key movement report history is an important fact in any organization security system. The security department will use the key movement history to keep track of staff and keys for a particular day and time. This report will help the security department in staging a good investigation when the need arises.

The manual generation of this investigated report is usually tedious, because the security staff has to open many pages of the KRB just to get an entire record for a department; hence the procedure is not appropriate in information gathering; and is highly time

consuming for a security officer. The new system therefore has database server that stores information and also allow easy access and retrieval of information when need arises.

REFERENCES

[1] Delgado, Ray (2004): Ethical controversies of peer-to-peer file sharing. Stanford Report, March 17, 2004.

[2] John Azzolini (2000). Introduction to Systems Engineering Practices. July 2000.

[3] Bruza, P. D., Van T. P., (1993); The Semantics of Data Flow Diagrams", University of Nijmegen.

[4] W. Stevens, G. Myers, L. Constantine, (1974); "Structured Design", IBM Systems Journal, 13 (2), 115-139.

[5] Chris G and Trish S. (1977); Structured Systems Analysis: Tools and Techniques. McDonnell Douglas Systems Integration Company.

[6] Martin F, Kendall S (2000): UML Distilled**,** Addison-Wesley USA

[7] Margaret H. D, Abdelsalam H,and Santosh B, (1977); A Mobile Transaction Model That Captures Both the Data and Movement Behavior. Mobile Networks and Applications. vol 2 no 2. pp. 149-162, 1997.

[8] Ebling M.R., Hunt GD. and Lei H. (2001); Issues for Context Services for Pervasive Computing. In Proc. Workshop on Middleware for Mobile Computing, IFIP/ACM Middleware 2001.

[9] Garlan D, Siewiorek D, Smailagic A, and Steenkiste P. (2002); Project Aura: Toward Distraction-Free Pervasive Computing IEEE Pervasive Computing, April-June 2002

[10] M. Haahr, R. Cunningham and V. Cahill. (2000); Towards a Generic Architecture for Mobile Object-Oriented Applications. SerP 2000: Workshop on Service Portability. San Francisco, December 2000.

[11] G.H. Kuenning, W. Ma, P. Reiher, and G.J. Popek (2002); Simplifying Automated Hoarding Methods. 5th ACM International Workshop on Modeling, Analysis and Simulation of Wireless and Mobile Systems SWiM 2002), Atlanta, GA, September, 2002.