ONLINE STUDENT EXEAT SYSTEM (OSES)

AYOKHAI, JOSHUA OKHAI
15010301007

BEING A PROJECT SUBMITTED IN THE DEPARTMENT OF COMPUTER SCIENCE AND MATHEMATICS, COLLEGE OF BASIC AND APPLIED SCIENCES IN PARTIAL FUFILLMENT OF THE REQUIREMENTS FOR THE AWARD OF DEGREE OF BACHELOR OF SCIENCE MOUNTAIN TOP UNIVERSITY, IBAFO, OGUN STATE, NIGERIA

2019

CERTIFICATION

This Project titled, **ONLINE STUDENT EXEAT SYSTEM (OSES),** prepared and submitted by **AYOKHAI, JOSHUA OKHAI** in partial fulfilment of the requirements for the degree of **BACHELOR OF SCIENCE (Computer Science),** is hereby accepted

____________________________ (Signature and Date)
Dr (Mrs.) O. O. Olaniyan
Supervisor

____________________________ (Signature and Date)
Dr I. O. Akinyemi
Head of Department

Accepted as partial fulfilment of the requirements for the degree of BACHELOR OF SCIENCE (Computer Science)

____________________________ **(Signature and Date)**

Prof. A. I. Akinwande
Dean, College of Basic and Applied Sciences

DEDICATION

This project is dedicated to God Almighty the giver of life and wisdom.

ACKNOWLEDGEMENT

I would like to acknowledge the sustaining power of the Almighty God for wisdom and understanding.
I would also like to acknowledge the Dean of the College of Basic and Applied Sciences Professor A. I. Akinwande for his leadership and support, my Head of Department Dr I. O. Akinyemi, for his support and guidance throughout my programme and the effort of my supervisor Dr. (Mrs.) O. O. Olaniyan for taking the pains out of no time to attend to me.

I acknowledge the effort of all the academic and non-academic staff of the Department of Computer Science and Mathematics for making my stay a worthwhile one Dr M. O. Oyetunji, Mr O. J. Falana, Dr. (Mrs.) O. O. Olaniyan Dr. O. B. Alaba, Dr P. A. Idowu, Dr O. A. Ojesanmi, Dr F. A. Kasali, Dr F. O. Oladeji, Mr. M. O. Okone, Mrs O. Taiwo and Mr. O. A. Taiwo.

Finally, I would like to appreciate my parents and my siblings for their love, support and care, and my friends for being there for me. I say God bless you richly.

ABSTRACT

This project is an online exeat system developed for the Mountain Top University. The basic problems facing exeat application and processing is that there is no system or database that keep the record of the exeat taken by students. Also, the process is time consuming and sometimes tedious, thus, there is a need to design and implement an exeat system. The aim of this project is to create a database that stores the information about the exeats of students and develop an online student exeat system for the use of Student Affairs Division of Mountain Top University.

This project is implemented using the Rapid Application Development Model for system development. The problems facing the current system were analyzed and the following need were identified which needed to be met. In prototyping, several versions of this solution were made. During testing, the application's implementation was checked to see if all the designs and ideas are viable and to better adjust and fine-tune designs and features in the project.

This project was finally completed and implemented by the use of C sharp an. Microsoft Visual Studio and Microsoft SQL server, and was run on a local server.

This project was done to develop an online student exeat system for the Mountain Top University for the purpose of exeat application and management. The recommendation would be the implementation and adaptation of this project onto their current system in order to improve the management and information processing of their institution.

KEYWORDS: Exeat System, Exeat Processing, Exeat, Student Affairs, Web Application

TABLE OF CONTENT

Content Page

Content Page

LIST OF FIGURES

LIST OF TABLES

CHAPTER ONE

INTRODUCTION

1.1 Background to the Study

The word ‘exeat’ is a general term used commonly to portray a period of absence from a Centre of learning either for a day, or parts of a day for appointments, or other reasons in privately owned academic environment or setting (Frischholz & Dieckmann, 2000). The current surge in computer technology and its various and diverse applications in every aspect of society causes creative, effective and confident use of Information and Communication Technology (ICT) an essential and required skill for everyday living. ICT capability encompasses not only the mastery of technical skills and techniques, it also facilitates the understanding of these skills in learning, everyday life and employment. ICT capabilities are fundamental to participation and engagement in modern society (Dutta, 2009). The idea of exeat is also used in certain learning institutions to describe a note needed to take leave from the college or university either for whole days or sections of a day for meetings, interviews, open days and other fixtures. The concern of access control is to determine permitted activity of valid users, arbitrating any attempt by a user to access a resource in the system. Several ways and techniques have been introduced to limit access to different fields of human activities (Omidiora, Olabiyisi, Arulogun, Oyeleye, & Adegbola, 2009), but little has been achieved about the exeat processing system in the private scholarly domain, such as the Mountain Top University. The current manual file processing approach to exeat has been found to be inert and in need of overhaul because it is not secure and can easily be duplicated or faked and does not provide a proper and reliable student monitoring.

Today, growth in academics can be of various concerns. Academics today is made possible by many structures and systems put in place to help simplify and achieve the goal of teaching and research. One of these structures that cannot be overlooked is that of inter-personal relationships between the students, the staff (whether academic or non-academic) and the management of the academy of learning institution. Another issue we which must also not fail to notice is that of exeat application and processing. For effective education to be rendered, there are some issues in the academic environment that have to be properly addressed; a particular instance is the issue of the exeat processing system in the university. These issues have thus created problems which inhibit academic furtherance in various aspects if academic growth and advancement. To support this proposed approach, this project identifies a range of options that can be used to manage and resolve exeat application and processing. This thus includes, at a point where the opportunity presents itself, the essential need for an administrator to appropriate every effort to resolve potential or actual academic complaints as thoroughly as possible, hence, the need for this project.

1.1 Statement of the Problem

Design and implementation of the Online Student Exeat System is a web-based application that is well needed in our modern society, and would solve some of the problems students face in the university environment. The basic problems facing exeat application and processing is that there is no system or database that keep the record of the exeat taken by students, as such no adequate security measure is provided. Also, the process is time consuming and sometimes tedious. Therefore, there is a need to design and implement an exeat system that would help monitor and keep records of students that take exeat to leave the school for one reason or the other.

1.2 Aim and Objectives of the Study

The aim of this project is to design and implement an online student exeat system for the use of Student Affairs Division of Mountain Top University, Ogun State. The specific objectives are to:

i. create a database that stores the information about the exeats of students
ii. develop an online student exeat system for the use of Student Affairs Division of Mountain Top University.

1.3 Scope of the Study

This study is focused on the design and implementation of an online exeat system majorly for the use of Student Affairs Division (SAD) of Mountain Top University, Ibafo, Ogun State. This system would be used by the SAD to store exeat records of students and number of times the students request for permission to leave the school premises.

1.4 Significance of the Study

This study would be of better benefit to Mountain Top University than the existing or current system in place which is manualy oriented. This is as a result of a proper database that has an effective and efficient way to store and manage the records of students. The system would easy to use by the students at the time of application for exeat anytime and from anywhere. Nevertheless, the Staff and Management can equally respond to student exeat by approving or declining the exeat.

The study would also be significant to other institutions who also make room for exeat giving to their students while on school campus. The developed application would be useful to other researcher who would find the codes of this system useful as it would save time and cost of coding.

1.5 Definition of Terms

Academic: relating to schools, colleges and universities, or connected with studying and thinking, not with practical skills.

College: Any place for specialized education after the age of 16 where people study or train to get knowledge and/or skills.

Lecturer: Someone who teaches at a college or university.

Procedure: An established or correct method of doing something.

Registration: The process of enrolling at a college or university, choosing courses, and paying fees at the beginning of an academic term.

Staff: the group of people who work for an organization.

Tedious: Boring because of being long, monotonous, or repetitive.

1.6 Organization of Subsequent Chapters

The chapters proceeding this chapter entail:
Chapter two - which gives a literature review on past and existing projects in the same are of exeat systems and their management, the exeat system, its need and its importance.
Chapter three – gives an explanation of the systems development, modules and methodologies, as well as the technologies involved.
Chapter four – gives an explanation of the completed system and the outputs.
Chapter five – this gives a summary of the entire project and well recommendations regarding it.

CHAPTER TWO

LITERATURE REVIEW

2.0 Introduction

An exeat typically means a written slip or piece of paper, authorizing permission for a particular student to leave a school or college for a period of absence from a centre of learning either for entire day, or parts of a day for appointments, interviews or other stated reasons from a privately-owned academic environment. This section cover and overview of the exeat system, the need for an exeat system and a review of relevant literatures and material that give insight on the subject.

2.1 Overview of Exeat System

The notion of exeat is used commonly for the description of a period of absenteeism from a centre or institution of learning. It is also used at certain colleges to define a mandatory note to take absence from the institution or school for either an entire day, or parts of a day for appointments, interviews, open days and other fixtures or schedules (Frischholz & Dieckmann, 2000). Access control or movement control is concerned with identifying allowed activities of authorized users, facilitating every attempt by a user to access a resource in the system. Several measures and methods have been put in place to control access to various paradigms of human activities (Omidiora, 2009), but little has been done with regards to exeat monitoring system in the privately-owned academic institution known as in the Mountain Top University Ibafo, Nigeria. The current paper-based file processing system approach to exeat has been found to be inefficient because it can be very slow and time consuming and does not provide a reliable student monitoring and management solution.

2.2 The Exeat System

The term exeat usually denotes a written slip or piece of paper, authorizing permission for a particular student to leave a school or college for a period of absence from a centre of learning either for entire day, or parts of a day for appointments, interviews or other stated reasons from a privately-owned academic environment. The current method of monitoring student’s movement is stressful and brings difficulty to the university management process in checking and safeguarding students’ exit/entry into the halls of residence.

2.3 Need for Exeat System

Because of the ever-present challenges faced in the simple problem of filing and managing exeats, and keeping track of students on and off campus, an automated and computerized solution must be put in place to address the challenges faced. In the article 'Anytime anywhere-remote monitoring of attendance system based on RFID using GSM network' the writers established that digital systems can basically solve the problems in the traditional system and thus increase the efficiency of equipment management and maintenance (Singhal & Gujral 2012).

2.4 Automating the Exeat Process

The automation of the exeat system is a feat that has been attempted over the past few years by different researches for simplification and ease of management. The principals involved in the automation of the exeat system very closely mirrors that of the leave management system. Projects like the Students Exeat Monitoring System Using Fingerprint Biometric Authentication and Mobile Short Message Service (Olaniyi, 2012) and Leave Master (Leave Master, 2014) are similar examples of systems with attempts to simplify the process and improve its efficiency. The automation of the exeat system would be done with the use of various computer technologies like computer programming and use of the internet and computer hardware.

2.5 Review of Relevant Literature

In 2014, an article on the design and implementation of an electronic exeat system was released in the International Journal of Scientific and Engineering Research. The project encompassed several principles in the overall design of an exeat and gave in-dept insight on the exeat application and approval process in use in the institution of its application. The author consequently proposed that the system would 'ensure that unauthorized exits are prevented by the use of biometric verification' and took several measures to put that in place. In the same article, by Onuiri, et al., (2014) it was outlined that the system would require all students being registered biometrically, which was the main base for security for the project. The was designed to also generate an approximation of the total number of students on and off campus, at every given point in time by implementing the following:

i. providing an efficient and more effective method of exeat processing/scheduling.
ii. defining a secured method of authorizing exit of individuals.

iii. confirming that students' residence and exeat details would be registered, to exit campus.

The above system was built with the particular needs of its case institution in mind and factors particular to that institution were addressed.

Another material of interest that of Olaniyi, Omotosho, Oluwatosin, Adegoke & Akinmukomi (2012). The project 'Students Exeat Monitoring System Using Fingerprint Biometric Authentication and Mobile Short Message Service' was done with the area of SMS (Short Message Service) technology as its main basis for communication among parties. The infrastructural model for the system applied similar technologies like: Microsoft Visual Studio (.Net Framework), C-sharp programming language, SQL programming language and Microsoft SQL Server 2005 for its software development needs and other external infrastructures like: finger print scanner, database server, SMS gateway and a biometric authentication framework for the complete implementation and success of the project. The project was designed to signal a parent or guardian when their child or ward was scheduled to leave the institution premises by locating the details of the student through their fingerprint biometric data and then sending the details of the exeat application or schedule to the parent or guardian through the use of and SMS. The system was automated by the use of these technologies and was tested using actual students' data in a typical university setting. The system could in turn manage students who were out of school premises because it kept track of those that had left and those who came back from their exeat were required to scan for a second time, signalling their arrival back on the university campus.

Other projects or solutions which performed similar tasks to this exeat system are:

i. Leave and Absence Management System (LAMS) – Durham University: is an online system for recording leave and absence of staffs in Durham University throughout all departments. The system just like this project was created to replace the previous and outdated paper-based system in use at the university. The system takes record leave of absence for staffs in their categories according to their designations and provides management information for proper staff management. The system helps process of leave scheduling, making it more efficient and economical for both staff and management respectively (Leave Master, 2014).

ii. Leave Master Software: this software solution is used for application of employee leave across different organizations. It shortens process time and streamlines the entire leave application and management process, eradicates the costly blunders and monitors all categories of employee leave and absenteeism including holidays, training and sick days and produces reports highlighting absence trends and patterns (Online Leave Management System, 2019).

iii. Boarding School Students Monitoring System (E-ID) Using Radio Frequency Identification: the application of RFID Matrix Card system as a monitoring system was to improve school management and monitor interest group movement. The system used main component of passive RFID system, database management system and wireless networking (Kadir, Wahab & Kanafiah, 2009).

iv. An SMS and RFID-Based Notification System of Lipa City Colleges, Lipa City, Batangas, Philippines: the researchers of this project conducted this system development study which helped the school to solve the problem in monitoring the students' whereabouts. It was aimed to design a system that would exploit the use of SMS module and RFID technology and also develop a system which will guarantee the privacy and security of information saved in a database (Mojares, Litan & Mojares, 2013).

2.6 Limitation of the Existing System

Below are stated key limitations to the current existing system in place in the University.

i. Lack of fitting security and upkeep of the complaint record in the system without proper cloud storage and record keeping.

ii. Having a manual system may lead file loss of damage while the application is in transit or even in storage. There is a lot of paper work resulting in slow data processing.

iii. Because the applications and files are only stored in paper, it is susceptible to damage and destruction by fire, water or any accident that may occur in the work place and no replacement record will be available. There is no system or database set up to screen transfer of complaint submitted on paper or as verbal representation.

iv. Time Consuming and very tedious.

v. All information is not placed separately thus, relevant data cannot be extracted from it. Thus, it is difficult to find records of past transactions due manual file management system.

CHAPTER THREE

METHODOLOGY

3.0 Introduction

This project is implemented using the Rapid Application Development Model for system development. This model breaks down the development process into several phases (KissFlow, 2019).

1. Define the requirements
2. Prototype
3. Receive Feedback
4. Finalize Software

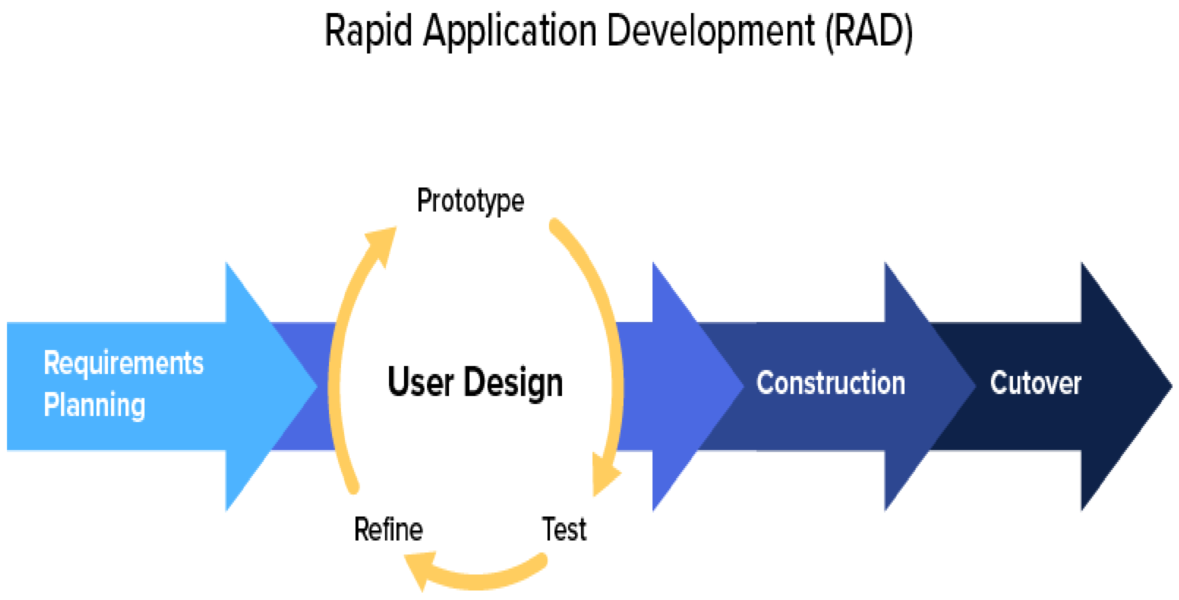


Figure 3.1. Rapid Application Development Life Cycle (KissFlow, 2019).

1. Define the requirements – the problems facing the current system were analyzed and the following need were identified which needed to be met. The requirements of this system include:
   1. An online implementation of a faster viable system to replace the current outdated one.
   2. A system which can monitor students who are out of school and those who have returned.
   3. A system which can allow for exeat application from anywhere and at any time.
   4. A system which exeat applications can be addressed irrespective of time and location.
   5. A system that would reduce the processing time for exeat applications.
2. Prototype – these are several versions of this solution. This involves multiple rounds of dummy projects to test features and factors in the project and to determine the best possible approach and method or attack to the problem at hand.
3. Receive Feedback – this is the information gotten from testing the applications implementation to see if all the designs and ideas are viable and to better adjust and fine-tune designs and features in the project.
4. Finalize Software – this is the point of project finalization and completion. The project is now ready to be launched and implemented to solve its designated problems.

3.1 Analysis of the Existing System

The Mountain Top University has a very simple but rather stressful. The exeat application is done manually on paper at the student affairs office.

When the Student Applies for the Exeat himself or in person, the process is as follows:

Step One: Form collection and form filling at the student affairs. At this initial phase, the student appears at the student affairs office and asks for an exeat form. The form is then filled with the necessary information i.e., Date of Exit, Date of Arrival, Full Name with Matriculation Number, College, Department, Hall of Residence, Room Number, Purpose (Official/Unofficial), Reason for Exeat and Student Signature with Date.

Step Two: The student takes the form back to their department in order to get a signature of approval from his/her Head of Department (HOD) or acting Head of Department (Ag. HOD) who if need be comment and endorse the exeat scheduling.

Step Three: The exeat form is then returned to the student affairs office for the Dean of Student Affairs to append his/her signature.

Step Four: The exeat form is then forwarded to the Office of the Vice Chancellor (V.C.), where the V.C. would then append his/her signature if he/she sees the reason tangible and acceptable. The Exeat is then returned to the student affairs office where the student would then pick it up for use.

The entire process would take a minimum of two days (2 days) at best with consecutive follow-up to be approved and returned to the student for use. This process is very tedious and time wasting. A copy of the exeat form is then retained by the student affairs office and filed in their records, leaving them with only a physical copy and no digital evidence for the exeat. If the information is ever to be retrieved, it would take some time as there would have to be a lot o sorting in order to locate the only copy or evidence of the exeat.

The system in use in this case study is the outdated manual File Processing System (FPS) wherein exeats are made by writing them down twice, then the student carries it around to the different offices in order to get it sign and as such may not meet any of the principal officers on-seat at arrival and thus the exeat is delayed. This system is an unnecessary waste of paper and resources when it could be done from anywhere on an online mobile platform. This system is time wasting, inefficient and slow. With this system, the students are eased of the stress of running around to find staff, and the entire process can happen faster, from anywhere, and not waste any stationary resource. The at the end of the cycle of the current system, the student checks back in at the student affairs and the process is complete. This new system will all the entire event to be tracked and monitored accurately and efficiently, with every action and decision properly documented and stored appropriately with excellent record keeping and ease of referencing.

3.2 Justification of the New System

The proposed system will serve to accomplish certain objectives and thus bring solutions to the problems identified in the current existing system and in this project. This newly proposed online system will achieve:

1. A secure and accurate upkeep of the exeat record in the system with proper cloud storage and record keeping.
2. A system and database set up to catalogue exeats submitted on paper or as verbal representation
3. Time saving and stress-free exeat application and processing.
4. Properly structured information from which relevant data can be mined and processed for study and data mining.
5. Less paper work resulting in faster data processing and efficient use of resources.
6. Ease of record location and record tracking.

3.3 Methodology

This project is programmed and implemented with the use and aid of the Microsoft Visual Studios 2017 alongside Microsoft SQL Server 2017.

3.3.1 C-sharp Programming Language

A developed by the company Microsoft, it is on this programming language that most of this entire is mostly built upon. It basically is the back-bone of this project and essentially most of the coding done is in this language.

3.3.2 ASP .NET Web Application (with MVC)

.NET is a developer platform that consists of instruments, programming languages and libraries to build a wide variety of apps. The base platform offers parts for all kinds of applications. This will enable user login systems with personal accounts that provide adequate security and privacy for user information and safeguard data stored in the Online Student Exeat System with individual authentication implemented. The Model View Controller (MVC) also makes the application compartmentalized for easy debugging, maintenance, upgrades and adding special features. The

MVC categories the web application into the model (in which the database models are defined, alongside the relationships and structures), the View (where all views and forms are set and defined with their functions and benefits) and at last the Controller (which determines the interaction and communication between the Model and View and facilitates the interaction between the entire application and the SQL support database to send data queries to the Microsoft SQL server database).

3.3.3 Microsoft SQL Server Express 2017

The SQL Server implemented in this Web application for storing all the data involved in tables, their formatting, schema and values for convenient access, manipulation and storage.

3.3.4 Standard Query Language (SQL)

This is a programming language implemented to facilitate the transmission of data and queries between the C-sharp code of the front-end of the application and the Microsoft SQL server. It is one of the most popular query languages in use today and is implemented all over the world because of its robust and extensive vocabulary.

3.4 Categories of Design

The planning and architecture of this project are thoroughly broken down, and subdivided into several categories. This decomposition is necessary in order to have a full grasp of the processes involved and measures taken to ensure its success. The design categories are divided as such:

i. Input Design - designs and plans with regard to user input and queries.

ii. Output Design - designs and plans with regard to the application output and feedback.

iii. Database Design - designs and plans with regards to database schema, architecture and overall inter-relation.

3.4.1 Interface Design

This encompasses the designs for all the interfaces utilized in the application. These interfaces are divided into different groups as there will be different types of users with different needs, functions and responsibilities. The types of user are:

1. Student User – this user is the student. This user has interfaces for:
    i. Login – containing two texts boxes for username and password and a login button.
    ii. Dashboard - containing list of all actions.
    iii. Exeat Application
    iv. List of Exeats applied for
    v. Print Exeat

2. Departmental User – this user is the Head of Department. This user has interfaces for:
    i. Login – containing two texts boxes for username and password and a login button.
    ii. Dashboard - containing list of all actions.
    iii. Exeat Approval
    iv. List of Exeats Pending and Approved
3. Student Affairs/Vice Chancellor User – this user is a Student Affairs officer or the Vice-Chancellor. This user has interfaces for:
    i. Login – containing two texts boxes for username and password and a login button
    ii. Dashboard - containing list of all actions.
    iii. Exeat Approval
    iv. List of Exeats Pending and Approved

3.4.2 Output Design

This encompasses the designs for all the output for the system. This output is limited to the exeat completed exeat form to be printed by the student upon approval of the exeat application. It is to contain information regarding:

i. The student's name
ii. Department
iii. Matriculation Number
iv. Department

v. College/Faculty

vi. Name of Head of Department and Date of Approval

vii. Name of Student Affairs Officer who approved the Exeat and date of approval or the Vice Chancellors name and date of approval.

3.5.3 Database Design

1. Exeat Model Design

Exeat	necessity	Data Type	
Id	Auto Gen.	string	(PK)
official	Required	bool	
Title	Required		
Body	Required		
Depart	Required		
Arrive	Required		
ExeatType	Required		
HODStatus		int	
SAStatus			
Type_Name	required	bool	
user_matric_no	required	string	(FK)

Table 3.5.3.1 Exeat Model Design

The above text is the design of the Exeat Model without Departmental and Student Affairs/Vice Chancellor’s Approval. It shows the Id as the Primary Key and user_matric_no as a foreign key for the Student Model and Exeat Type Models respectively.

2. Exeat Type Model Design

Exeat Type	necessity	Data Type	
Name	required	string	PK

Table 3.5.3.2 Exeat Type Model Design

3. Student Model Design

Student	necessity	Data Type	
matric_no	Auto Gen.	string	PK
first_name	required	string	
last_name	Auto Gen.	string	
othername	required	string	
room_no	required	int	
HallofReside	required	string	FK
Programme	required	string	FK
HType_Name	required	date	
Type_Name	optional	date	
user_id	Required	int	FK

Table 3.4.3.3 Student Model Design

4. Hall of Residence Model Design

HALL OF RESIDENCE	necessity	Data Type	length	
Name	required	string		PK

Table 3.4.3.4 Hall of Residence Type Model Design

5. Programme Model Design

Programme	necessity	Data Type	length	
Name	required	string		PK

Table 3.4.3.5 Programme Model Design

6. AspNetUsers Model Design

AspNetUsers	necessity	Data Type	
Id	required	string	PK
Email	required	string	
EmailConfirmed	required	string	
PasswordHash	required	string	
SecurityStamp		string	
PhoneNumber		int	
PhoneNumberConfirmed		string	
TwoFactorEnabled		string	
LockoutEndDateUtc			
LockoutEabled			
AccessFailedCount			
UserName	required		

Table 3.4.3.6 AspNetUsers Model Design

7. AspNetRoles Model Design

AspNetRoles	necessity	Data Type	
Id	required	string	PK
Name	required	string	

Table 3.4.3.7 User Type Model Design

8. AspNetUserClaims Model Design

USER TYPE	necessity	Data Type	
Id	required	string	PK
UserId	required	string	
ClaimType		stringstring	
ClaimValue			

Table 3.4.3.8 AspNetUserClaims Model Design

3.5.4 Database UML

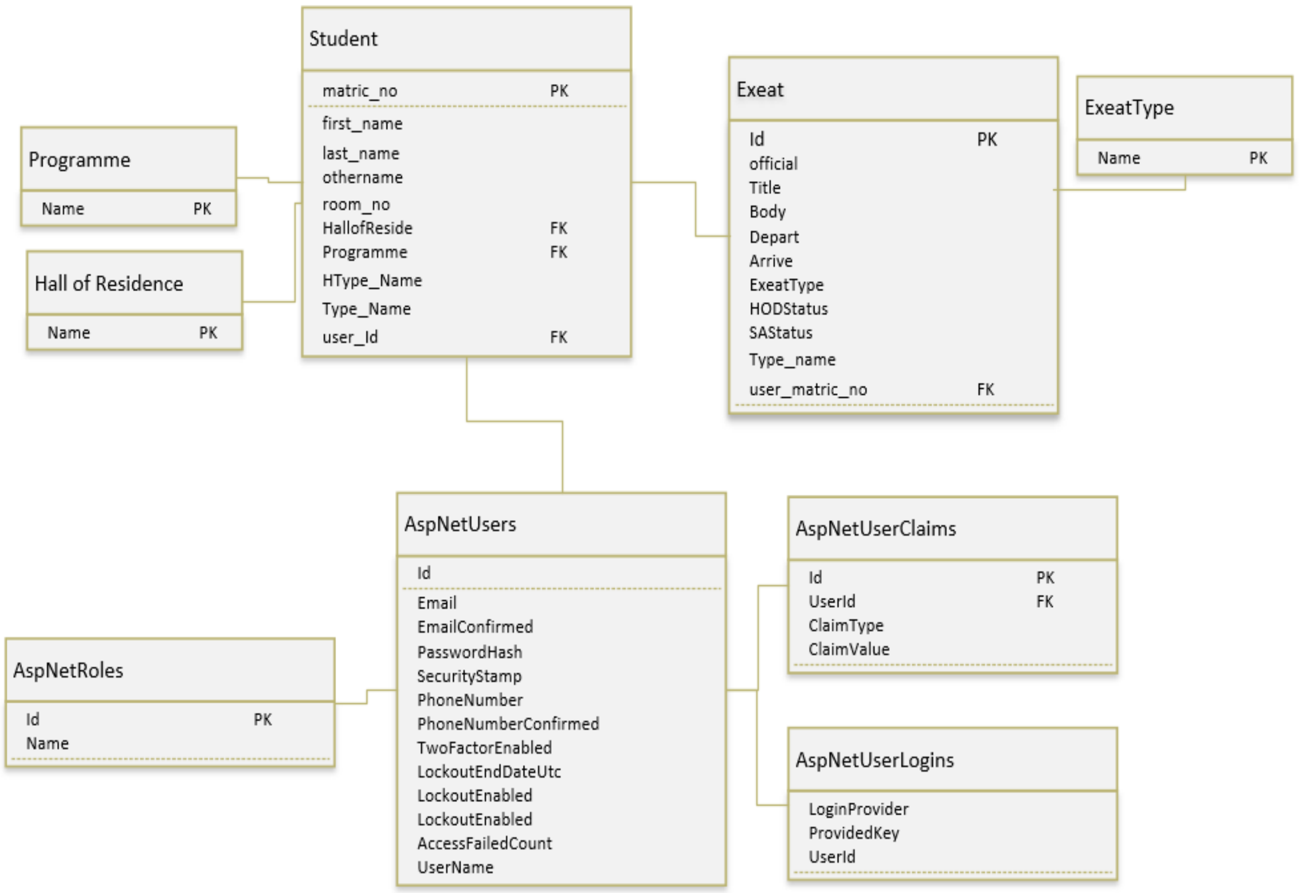


Figure 3.5.4.1 UML diagram illustrating the Database relations for the Online Student Exeat System.

3.6.1 System Modules

Login/Authentication Module: in this module, whichever user logs in will be logged into their own profile with their profile type. In this case, the username and password used for authentication will be supplied and when the system confirms its accuracy, the user is logged in.

Edit Password Module: This is available to the user at any time when logging in with the default password allocated by the system, or when his/her password is restored to the default password by the administrator upon the user's request.

Exeat Application Module: This allows a student to either schedule for a long-term exeat or short-term exeat.

All Exeats Module: Depending on the user type, it displays all the exeats the user has interacted with, that is, in the case of a student, all exeats allied for (whether past, pending or approved) either students may also view personal exeat history on the web application.

Logout Module: The user after exeat processing is expected to log out here.

Dashboard Interface: This application being a web application will be accessed through the internet. All users of this interface as regards will have it tailored to their user type. Its modules generic modules include: the login module, the edit password module, all exeats module, search module. Others as per user type include: for the student user – exeat application module, exeat printing module; for the Heads of Departments, Student Affairs Officer and Vice Chancellor – approve exeats. The view and process exeat module, the search module and the logout module:

Reset Password Module: Here, the administrator may reset to default his/her password or the password of a user as requested.

Register a Student Module: This module is for the registration of students as required. The student table would be populated from the school's record of students registered that semester on campus, thus, this module is present for likelihoods in the occurrence of hall of residence change or in case of an error in terms of student's hall as registered in the school's database.

Exeats Processing Module: This module is the heart of this interface. The Head of Department/Dean of Students Affairs or the Vice Chancellor may view and sort exeats at his/her convenience to make it easier to analyse exeats and authenticate. All exeats irrespective of type may be viewed based on date or type. Also, all short exeats may be distinguished from the long exeats as well as official and unofficial exeats; all pending exeats may also be differentiated from permitted exeats or denied exeats and so on. The exeats may be viewed as a table or singularly with more details as would be seen in the user guide below. As each exeat is viewed an exeat history of that student under view for that semester is shown. Changes are also all automatically saved.

Search Module: this allows Head of Department/Dean of Students Affairs or the Vice Chancellor to search for exeat records, based on name or matriculation number.

Logout Module: After carrying out exeat application/authorization, every user is expected to logout here.

3.6.2 Use Case Diagrams

The use case diagrams depicted below show the different types of user and their interaction with the system showing the relationship between the users.

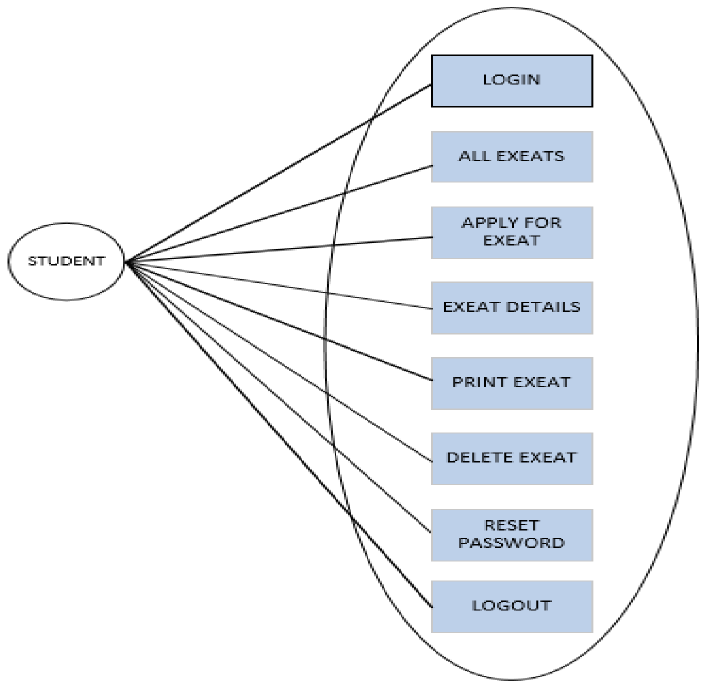


Figure 3.6.2.1 Use case diagram illustrating the Student User Processes

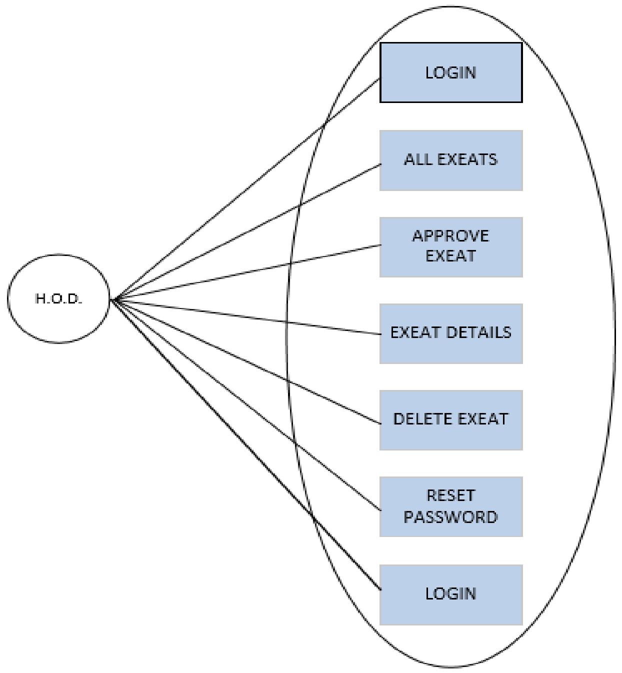


Figure 3.6.2.2 Use case diagram illustrating the Head of Department (H.O.D.) User Processes

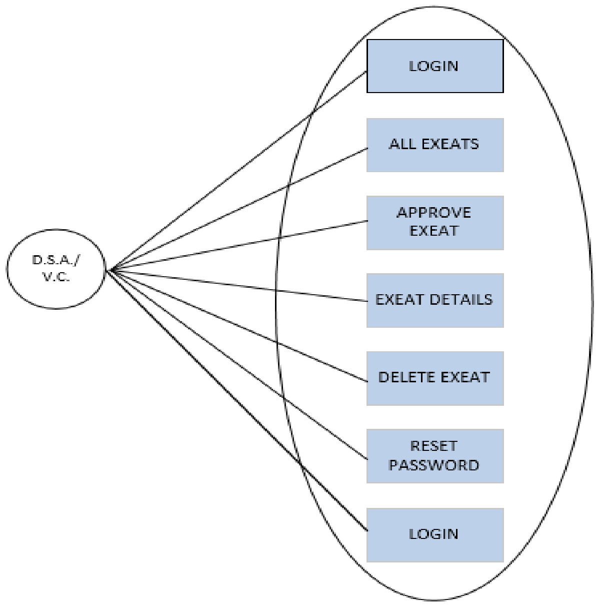


Figure 3.6.2.3 Use case diagram illustrating the Dean of Student Affairs (D.S.A.) / Vice Chancellor (V.C.) User Processes

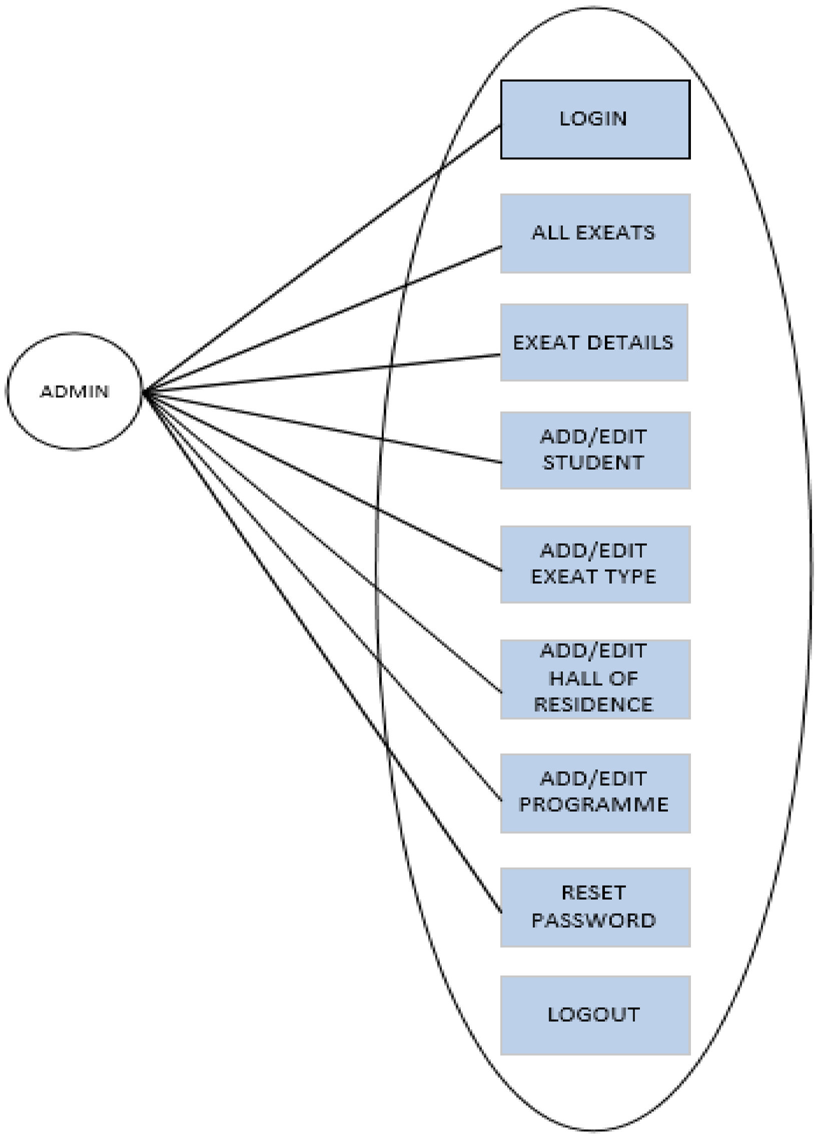


Figure 3.6.2.4 Use case diagram illustrating Administrator User Processes

3.6.3 System Flowchart

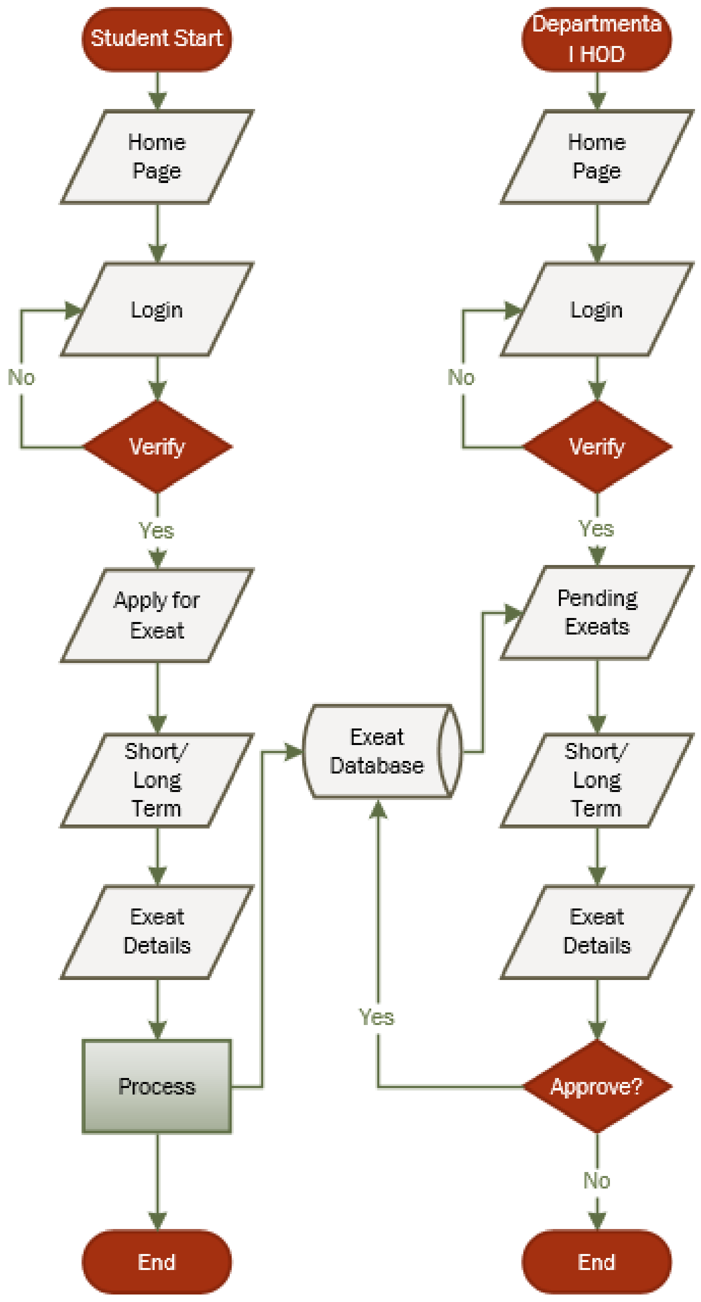


Figure 3.6.3.1 Flowchart illustrating the Student Exeat Application Process

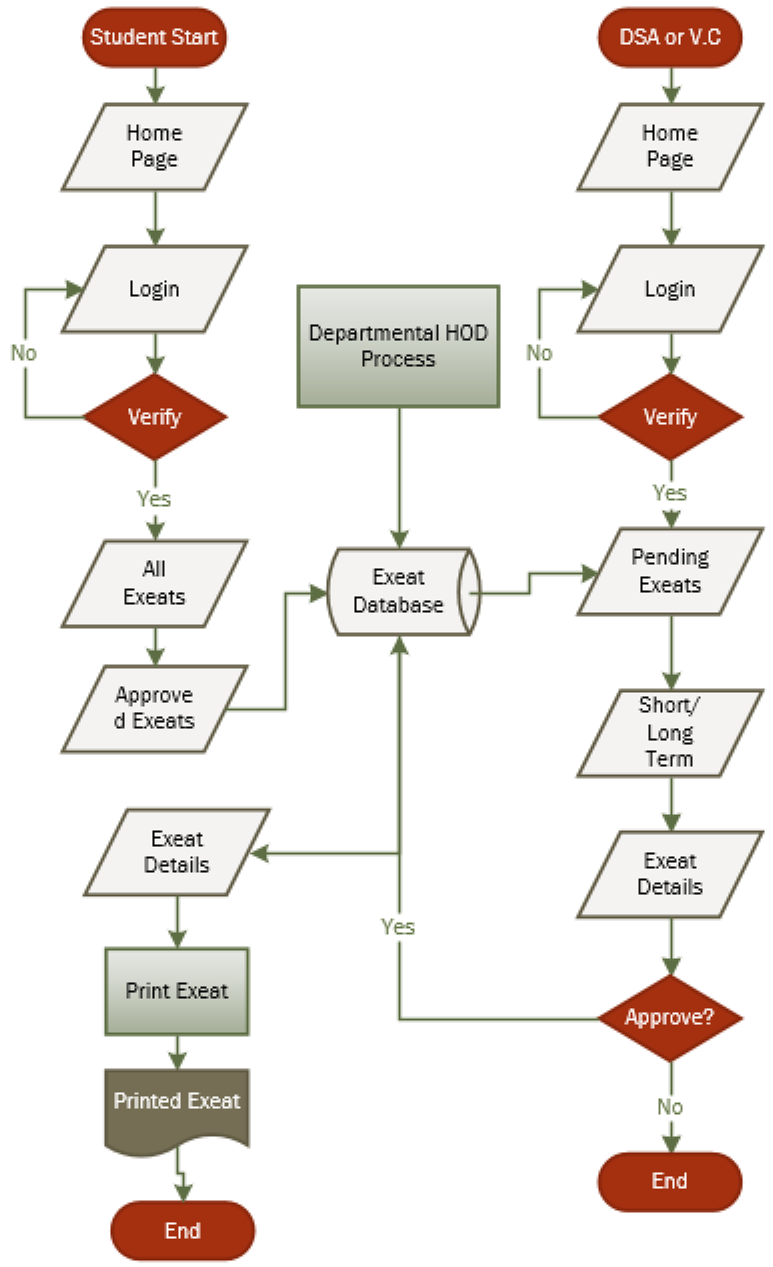


Figure 3.6.3.2 Showing the Final Approval Process and Exeat Printing Process.

This involves the Student Module, Department Module and Student Affairs and Vice Chancellor Approval Phase.

CHAPTER FOUR

DATA ANALYSIS, RESULTS AND DISCUSSION OF FINDINGS

4.0 Introduction

This chapter comprises of information pertaining to the resulting solution produced from the earlier stated technology involved in the implementation of the project and takes an in-dept look on their characteristics.

4.1 Choice of programming language and technology

The selected technologies and programming languages implemented are as follows.

4.1.1 C-sharp Programming Language

C# is a modern, general-purpose, object-oriented programming language developed by Microsoft and approved by European Computer Manufacturers Association (ECMA) and International Standards Organization (ISO).

C# was developed by Anders Hejlsberg and his team during the development of .Net Framework.

C# is designed for Common Language Infrastructure (CLI), which comprises of the executable code and the runtime environment which allows implementation of various high-level languages (H.L.L.) on diverse computer architectures and platforms.

The following reasons make C# a widely used professional language −

- Produces efficient programs.
- Can be compiled on a variety of computer platforms.
- Component oriented.
- Easy to learn.
- Part of .Net Framework.
- Modern, general-purpose programming language
- Object oriented.

- Structured language.

4.1.1.1 Features

C-sharp constructs closely follow conventional high-level languages (H.L.L) just like C++ and C, being an object-oriented programming language. It has strong similarities with Java, it has many strong programming features that make it endearing to a number of programmers worldwide.

Important features of C-sharp include but are not limited to Boolean Conditions, Automatic Garbage Collection, Standard Library, Assembly Versioning, Properties and Events, Delegates and Events Management, Easy-to-use Generics, Indexers, Conditional Compilation, Simple Multithreading, LINQ and Lambda Expressions and also Integration with Windows.

4.2.2 SQL

SQL is Structured Query Language, which is a computer language created for manipulating, storing and retrieving data stored/archived in a relational database.

SQL is a language to operate databases; it includes database creation, deletion, fetching rows, modifying rows, etc. SQL is an **ANSI** (American National Standards Institute) standard language, but many versions of the SQL language exist and are in use.

Relational Database System has SQL as its standard language. All the Relational Database Management Systems (RDMS) like MySQL, MS Access, Oracle, Sybase, Informix, Postgres and SQL Server use SQL as their standard database language.

SQL is widely popular because it offers the following advantages −

- Allows users to access data in the relational database management systems.
- Allows users to describe the data.
- Allows users to define the data in a database and manipulate that data.
- Allows to embed within other languages using SQL modules, libraries & pre-compilers.
- Allows users to create and drop databases and tables.

- Allows users to create view, stored procedure, functions in a database.
- Allows users to set permissions on tables, procedures and views.

4.2.1 Visual Studio 2017 IDE

Microsoft Visual Studio is an integrated development environment (IDE) from Microsoft. It is used to develop computer programs, as well as websites, web apps, web services and mobile apps. Visual Studio uses Microsoft software development platforms such as Windows API, Windows Forms, Windows Presentation Foundation, Windows Store and Microsoft Silverlight. It can produce both native code and managed code.

Visual Studio includes a code editor supporting IntelliSense (the code completion component) as well as code refactoring. The integrated debugger works both as a source-level debugger and a machine-level debugger. Other built-in tools include a code profiler, forms designer for building GUI applications, web designer, class designer, and database schema designer. It accepts plug-ins that enhance the functionality at almost every level—including adding support for source control systems (like Subversion and Git) and adding new toolsets like editors and visual designers for domain-specific languages or toolsets for other aspects of the software development lifecycle (like the Team Foundation Server client: Team Explorer).

Visual Studio supports 36 different programming languages and allows the code editor and debugger to support (to varying degrees) nearly any programming language, provided a language-specific service exists. Built-in languages include C, C++, C++/CLI, Visual Basic .NET, -sharp, F-sharp, JavaScript, TypeScript, XML, XSLT, HTML, and CSS. Support for other languages such as Python, Ruby, Node.js, and M among others is available via plug-ins. Java (and J#) were supported in the past.

4.2.2 Microsoft SQL Server

The Microsoft SQL Server created by Microsoft, is a relational database management system (RDBMS) that supports a wide variety of transaction processing, business intelligence and analytics applications in corporate IT environments. Microsoft SQL Server is one of the three market-leading database technologies, along with Oracle Database and IBM.

Like other RDBMS software, Microsoft SQL Server is built on top of SQL, a standardized programming language that database administrators (DBAs) and other IT professionals use to manage databases and query the data they contain. SQL Server is tied to Transact-SQL (T-SQL), an implementation of SQL from Microsoft that adds a set of proprietary programming extensions to the standard language.

Microsoft also bundles a variety of data management, business intelligence (BI) and analytics tools with SQL Server. In addition to the R Services and now Machine Learning Services technology that first appeared in SQL Server 2016, the data analysis offerings include SQL Server Analysis Services, an analytical engine that processes data for use in BI and data visualization applications, and SQL Server Reporting Services, which supports the creation and delivery of BI reports.

On the data management side, Microsoft SQL Server includes SQL Server Integration Services, SQL Server Data Quality Services and SQL Server Master Data Services. Also bundled with the DBMS are two sets of tools for DBAs and developers: SQL Server Data Tools, for use in developing databases, and SQL Server Management Studio, for use in deploying, monitoring and managing databases.

4.2.3 ASP .NET Technology

.NET is a developer platform that consists of instruments, programming languages and libraries to build a wide variety of apps. The base platform offers parts for all kinds of applications. Additional frameworks like ASP.NET extend. NET with parts to create particular kinds of applications.

4.3 The System Main Menu Implementation

The system was designed to have a Home Page, which serves as the Dashboard and has a menu of options which the user can choose in order to perform an action. Some of the pages only appear for role specific users, others are generic for access to anyone. Below is a table showing the links in the main menu, its navigation, their controller, links and role:

	Navigation	Controller	Link	Role
1	Home	HomeController	/Home	
2	About	HomeController	/About	
3	Contact	HomeController	/Contact	
4	Exeats	ExeatController	/Exeats	Admin, Student, HOD, DSA, VC
5	Hall of Residence	HallofResideController	/HallofResides	Admin
6	Exeat Type	ExeatTypeController	/ExeatTypes	Admin
7	Department	DepartmentController	/Departments	Admin
8	Students	StudentController	/Students	Admin, User
9	Login	AccountController	/Account/Login	

Table 4.3.1 Home Navigation Table.

4.4 System Testing

This is the main control measure used in the creation of software to guarantee service quality. The fundamental role is to identify software errors and guarantee that all requirements are adequately met. The computer software will then be performed and debugged after the coding stage. This means that the testing must not only reveal mistakes that were introduced during coding, but also mistakes that were implemented during the earlier stage. The objective of testing is therefore to uncover the program specifications, design and coding mistakes.

Objectives of testing

All objectives to this are clear and definite.

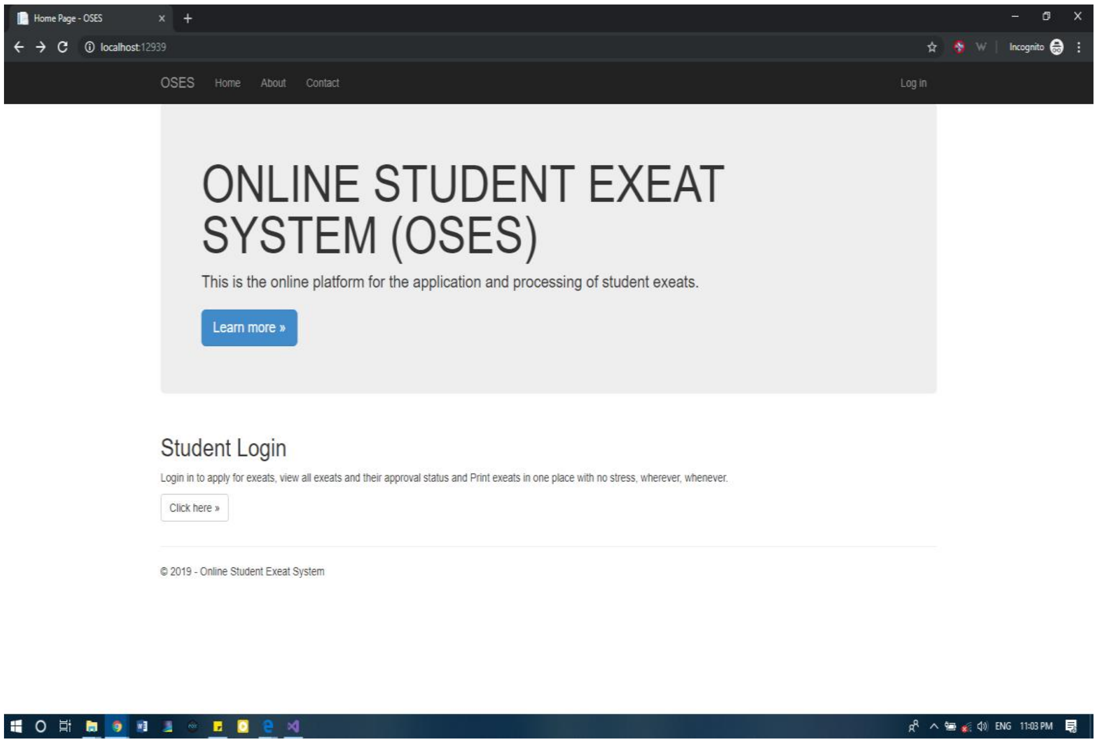
i. Executing the program with the intent of finding errors or problem at compilation.
ii. Using sample or dummy data in check data entry points and for functionality or faults.
iii. Testing the application module by module, allowing problems and faults to be pin pointed and located at the specific point where they may arise.

After counter to the commonly held view than a successful test is one in which no errors are found. In fact, our objective is to design tests that a systematically uncover different classes of errors and do so with a minimum amount of time and effort.

4.5 Test data

The test data used in the testing of this project is fictional and only valid for tests. All information is random and not real: the previewed exeat application cases a fictional, only generated for the testing phase of the project.

4.6 Snapshots of the System



ONLINE STUDENT EXEAT SYSTEM (OSES)

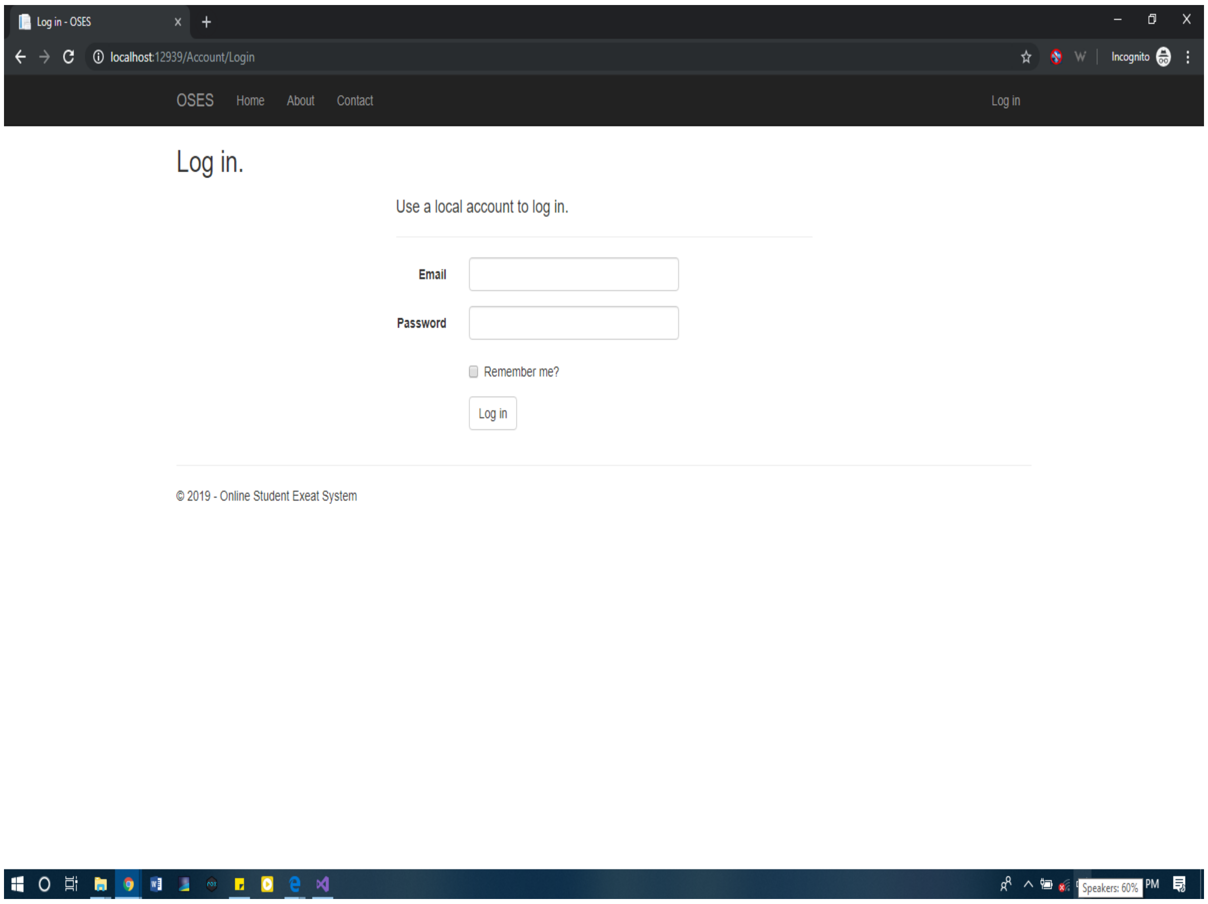
This is the online platform for the application and processing of student exeats.

Student Login

Login in to apply for exeats, view all exeats and their approval status and Print exeats in one place with no stress, wherever, whenever.

© 2019 - Online Student Exeat System

Figure 4.6.1 Home Page with Menu for Non-User.



Log in.

Use a local account to log in.

Email

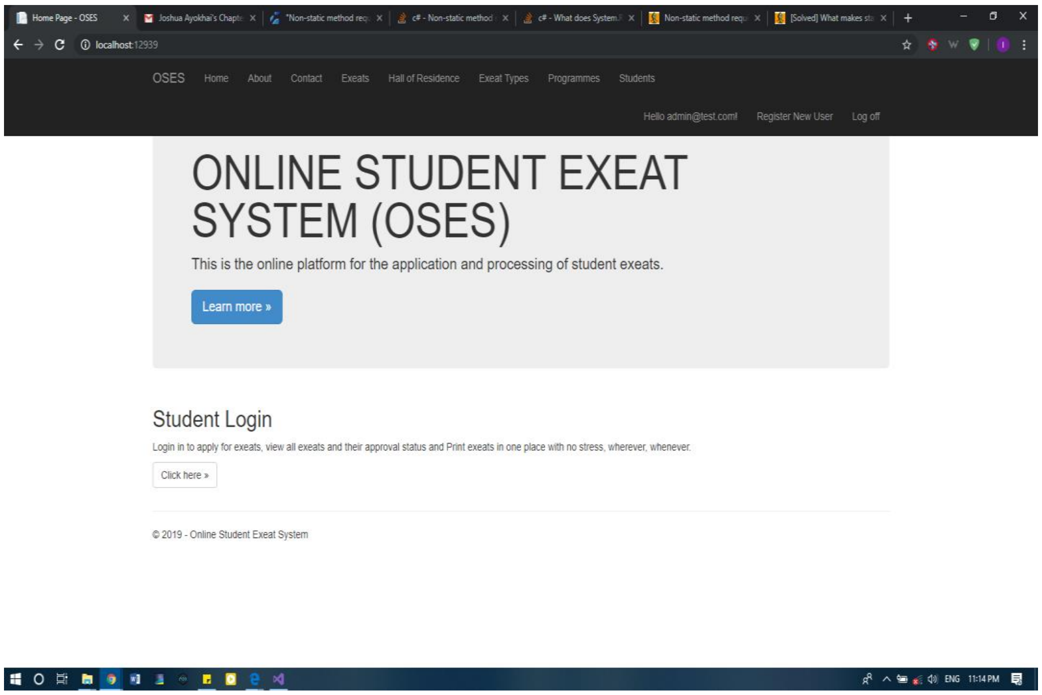
Password

Remember me?

Log in

© 2019 - Online Student Exeat System

Figure 4.6.2 Login Page for any user.



ONLINE STUDENT EXEAT SYSTEM (OSES)

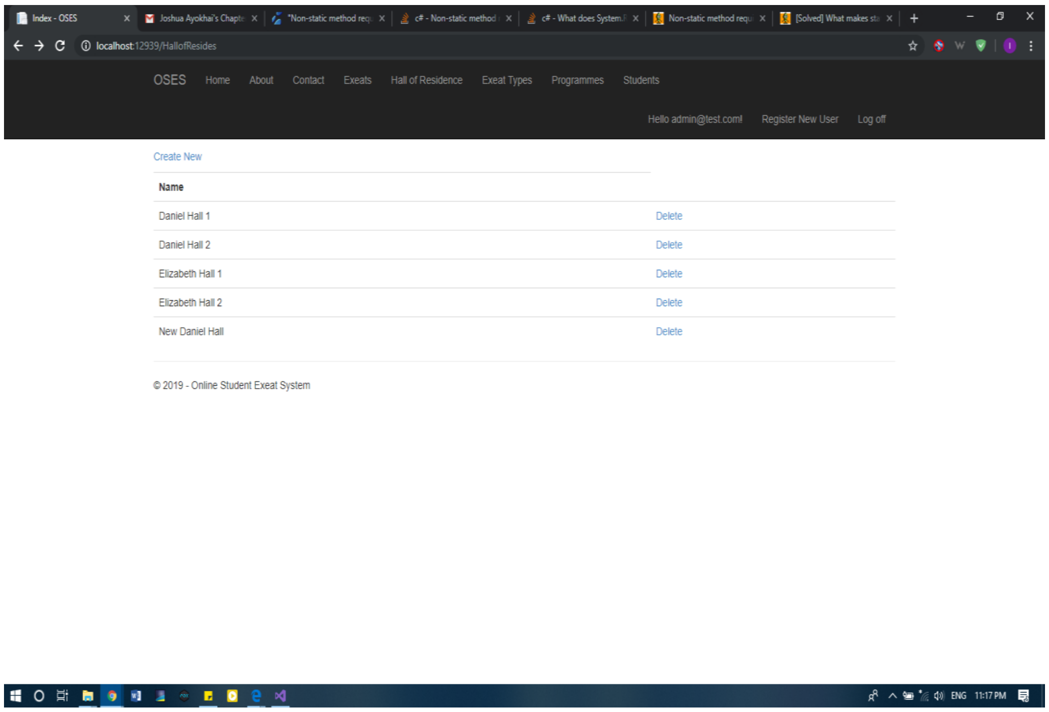
This is the online platform for the application and processing of student exeats.

Student Login

Login in to apply for exeats, view all exeats and their approval status and Print exeats in one place with no stress, wherever, whenever.

© 2019 - Online Student Exeat System

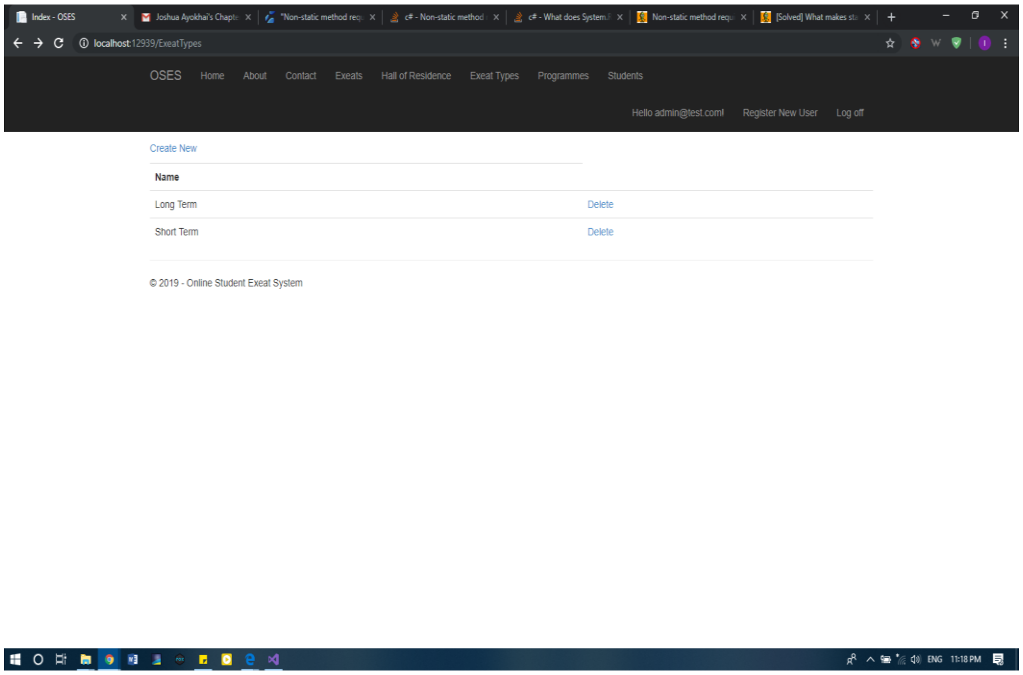
Figure 4.6.3 Home Page for Logged in Admin.



Name	
Daniel Hall 1	Delete
Daniel Hall 2	Delete
Elizabeth Hall 1	Delete
Elizabeth Hall 2	Delete
New Daniel Hall	Delete

© 2019 - Online Student Exeat System

Figure 4.6.4 Index Page for Hall of Residence (Admin)


OSES Home About Contact Exeats Hall of Residence Exeat Types Programmes Students

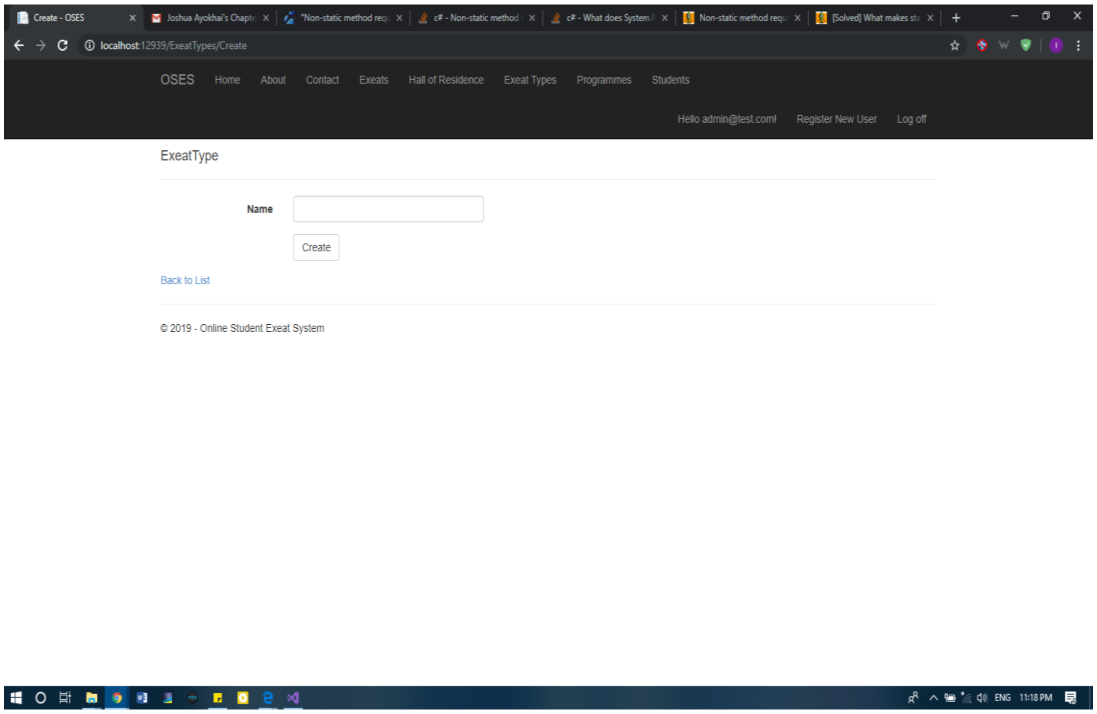
Hello admin@test.com! Register New User Log off

Create New

Name	
Long Term	Delete
Short Term	Delete

© 2019 - Online Student Exeat System

Figure 4.6.5 Index Page for Types of Exeats (Admin)


OSES Home About Contact Exeats Hall of Residence Exeat Types Programmes Students

Hello admin@test.com! Register New User Log off

ExeatType

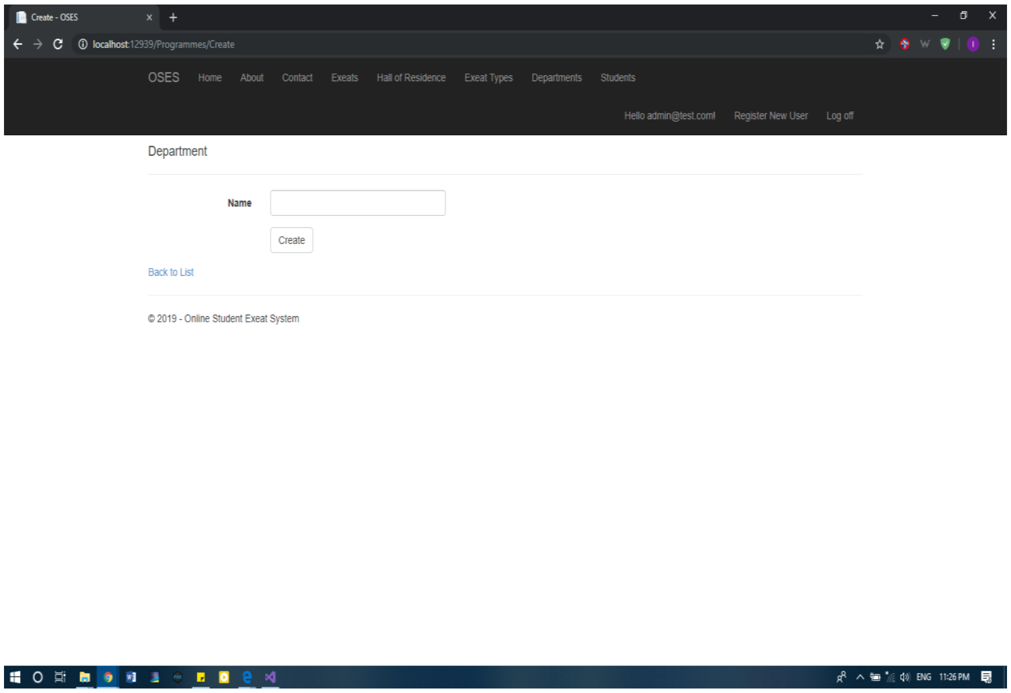
Name

Create

Back to List

© 2019 - Online Student Exeat System

Figure 4.6.6 Add Exeat Type Page (Admin)

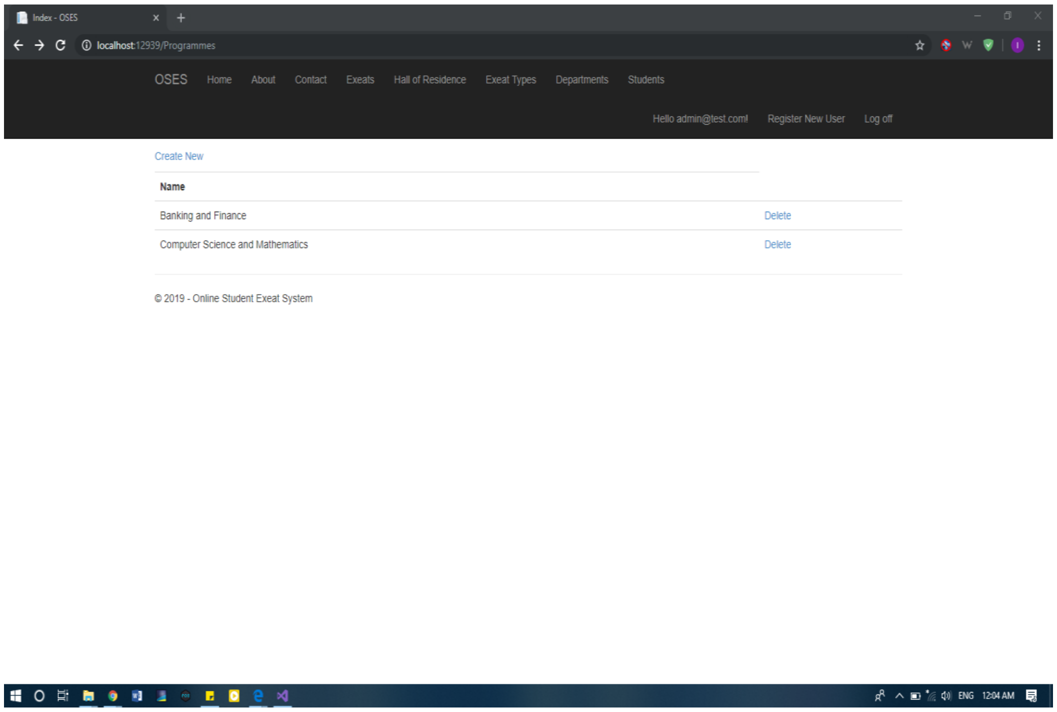


Department

Name	

© 2019 - Online Student Exeat System

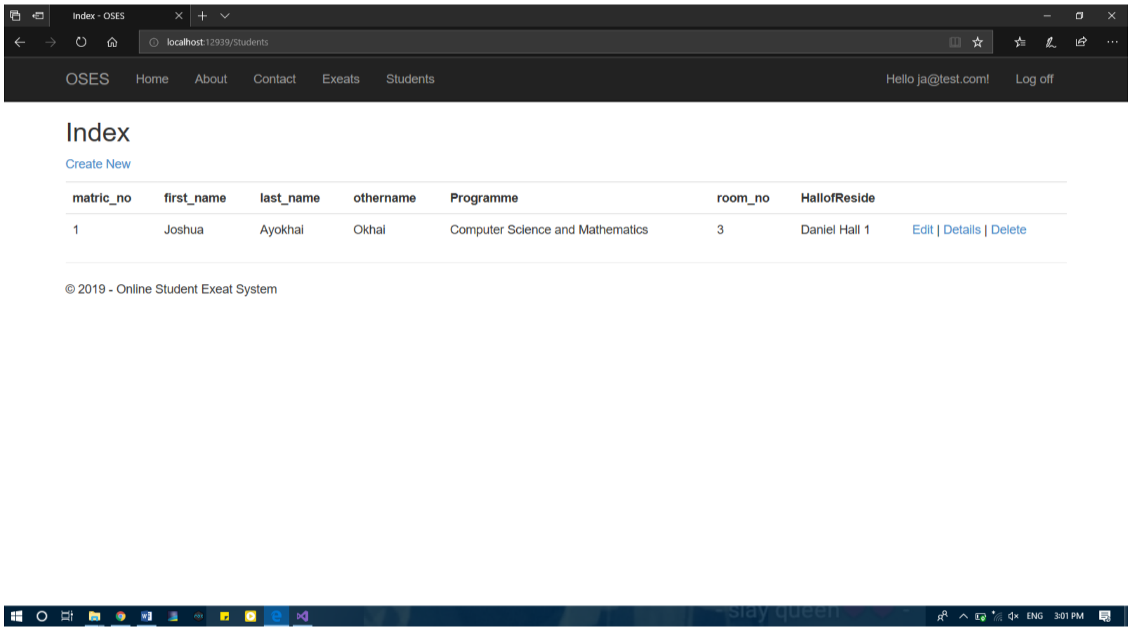
Figure 4.6.7 Add Department Page (Admin)



Name	
Banking and Finance	Delete
Computer Science and Mathematics	Delete

© 2019 - Online Student Exeat System

Figure 4.6.8 Index Page for Department (Admin)


Index - OSES
localhost:12939/Students
OSES Home About Contact Exeats Students
Hello ja@test.com! Log off

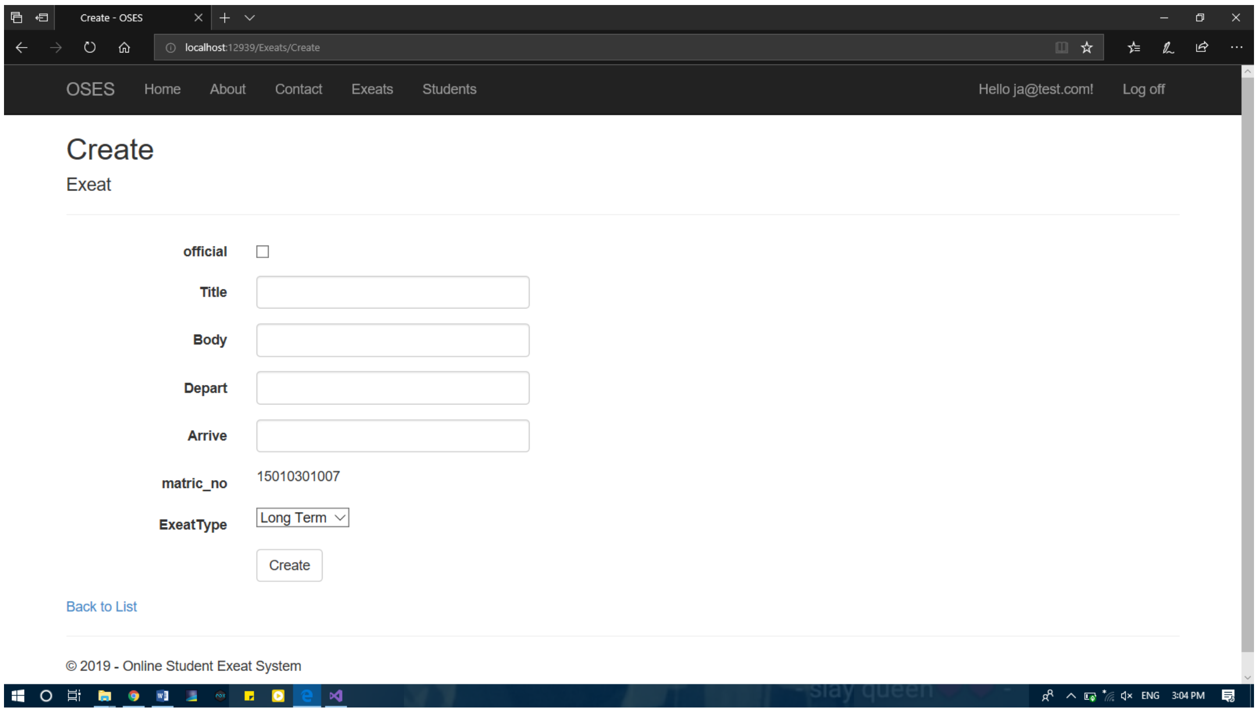
Index

Create New

matric_no	first_name	last_name	othername	Programme	room_no	HallofReside	
1	Joshua	Ayokhai	Okhai	Computer Science and Mathematics	3	Daniel Hall 1	Edit | Details | Delete

© 2019 - Online Student Exeat System

Figure 4.6.9 Index Page for Students (Admin)


Create - OSES
localhost:12939/Exeats/Create
OSES Home About Contact Exeats Students
Hello ja@test.com! Log off

Create

Exeat

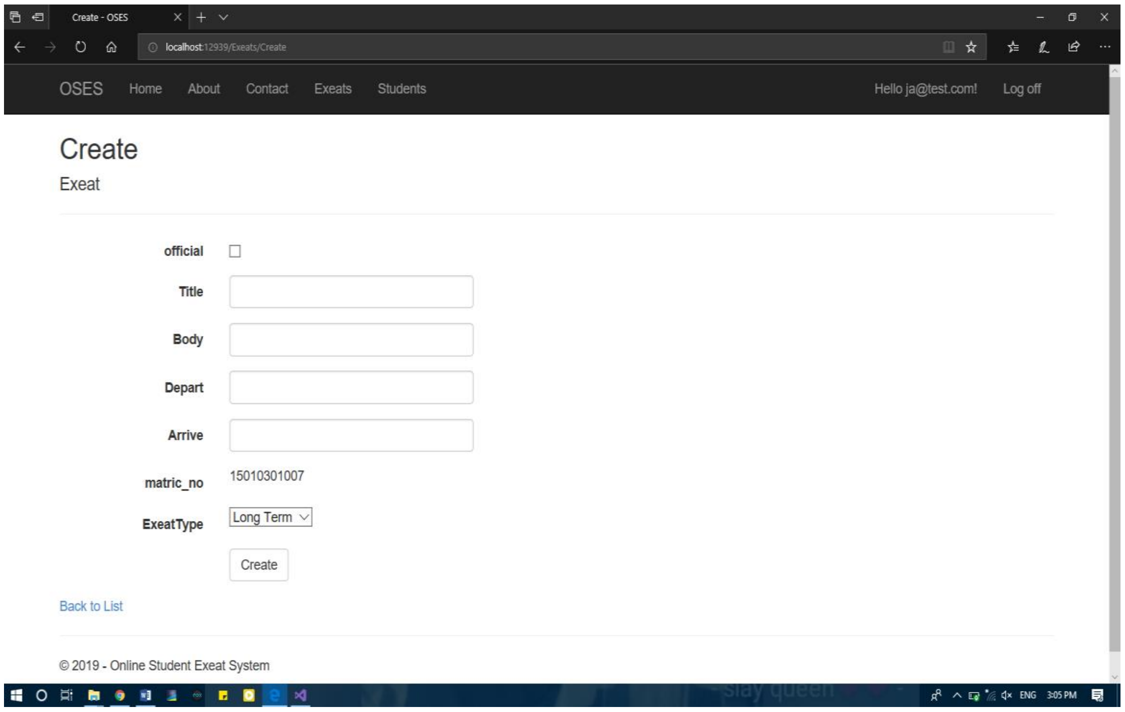
official	☐
Title	
Body	
Depart	
Arrive	
matric_no	15010301007
ExeatType	Long Term

Create

Back to List

© 2019 - Online Student Exeat System

Figure 4.6.10 Create Exeat Page (Student)


Create - OSES

localhost:12939/Exeats/Create

OSES Home About Contact Exeats Students Hello ja@test.com! Log off

Create

Exeat

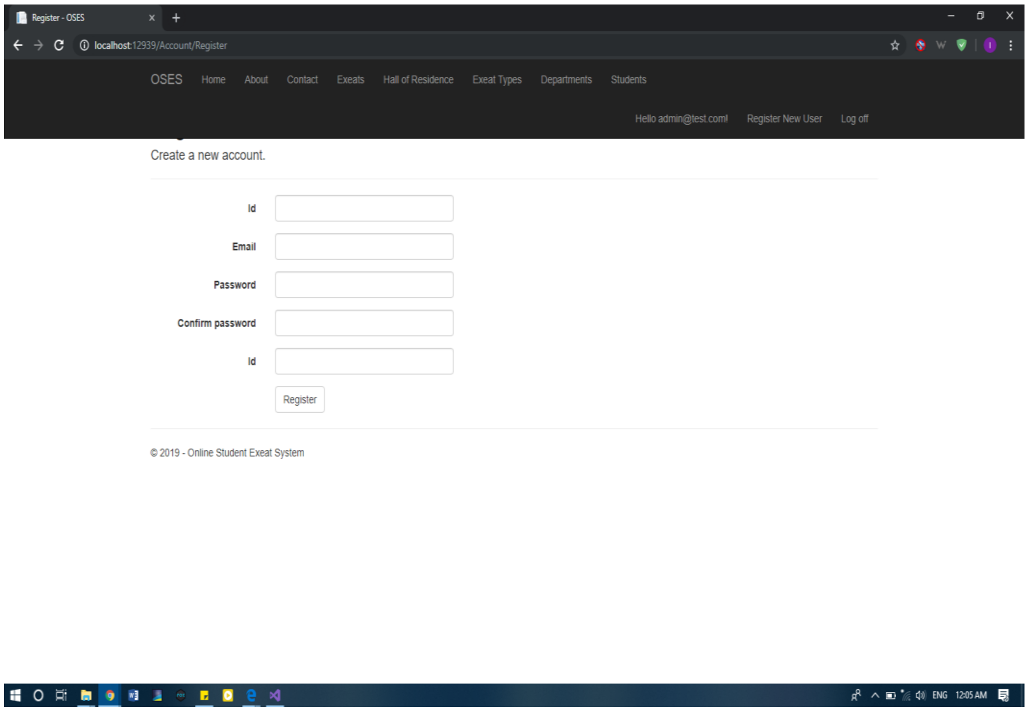
official	☐
Title	
Body	
Depart	
Arrive	
matric_no	15010301007
ExeatType	Long Term

Create

Back to List

© 2019 - Online Student Exeat System

Figure 4.6.11 Add Department Page (Admin)


Register - OSES

localhost:12939/Account/Register

OSES Home About Contact Exeats Hall of Residence Exeat Types Departments Students

Hello admin@test.com! Register New User Log off

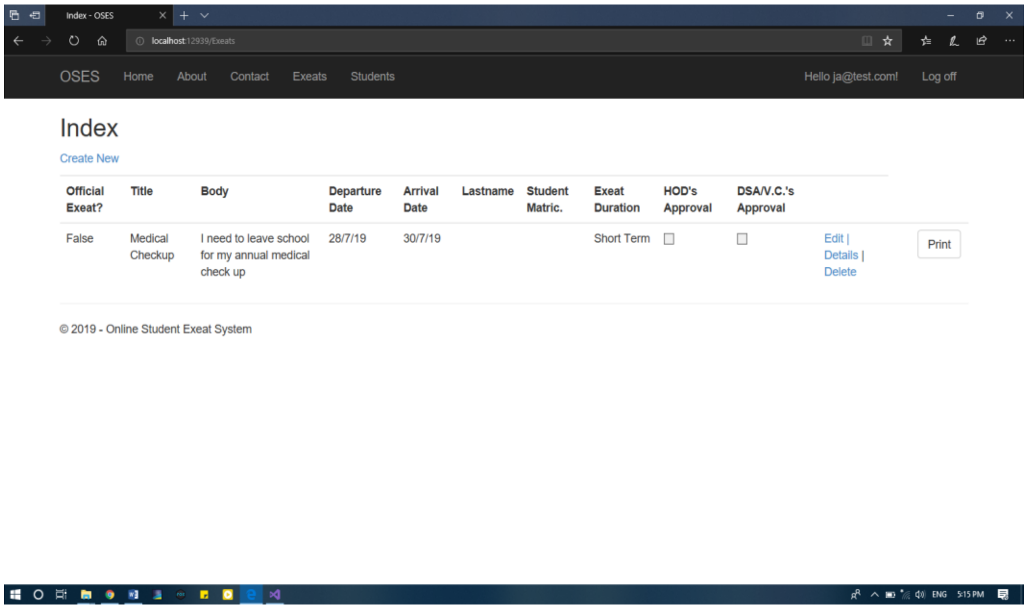
Create a new account.

Id	
Email	
Password	
Confirm password	
Id	

Register

© 2019 - Online Student Exeat System

Figure 4.6.12 Register New User Page.


Index - OSES

localhost:12939/Exeats

OSES Home About Contact Exeats Students Hello ja@test.com! Log off

Index

Create New

Official Exeat?	Title	Body	Departure Date	Arrival Date	Lastname	Student Matric.	Exeat Duration	HOD's Approval	DSA/V.C.'s Approval		
False	Medical Checkup	I need to leave school for my annual medical check up	28/7/19	30/7/19			Short Term	☐	☐	Edit | Details | Delete	Print

© 2019 - Online Student Exeat System

Figure 4.6.13 All Exeat (Exeat Index) Page (Admin, Students, H.O.D., D.S.A., V.C.).

CHAPTER FIVE
SUMMARY, CONCLUSION AND RECOMMENDATIONS

5.0 Introduction

This chapter consists of a comprehensive summary of the entire right up, findings in regard to the background study and implementation, suggestions for further work and lastly, a recommendation regarding the use and application of the solution produced in the course of this project.

5.1 Summary

A working system was developed with the Microsoft ASP .NET using Entity Framework and Bootstrap (for the frontend), and with use of Visual Studio 2017 and Microsoft SQL Server 2014. The project was implemented to have four categories of users:

i. Student
ii. Admin
iii. H.O.D.
iv. DSA/VC (Dean of Student Affairs or Vice Chancellor as the case may be).

The user types were designated different privileges in terms of access to data and queries. The Student user was allowed the login access and exeat creation and deletion and the ability to print his/her exeat, the Heads of Departments, Dean of Student Affairs and Vice Chancellor were given access to exeat records of their respective departments and access to all existing exeats respectively. Lastly, the administrator has complete control over the entire system and can add users and modify any user data existing on the system and can print any record within the system.

5.2 Limitation of Study

This project was completely implemented and completed, however, certain limitations were identified and caused major difficulties and delays. They are:

i. Scarcity of previous work
ii. Lack of privileges to access proper student data and scheme
iii. Lack of previously filed information for exeats

5.3 Recommendation

A strong recommendation to the management of the Mountain Top University and any other tertiary institution utilizing the exeat system for student movement, would be the implementation and adaptation of this project onto their current system in order to improve the management and information processing of their institution.

5.4 Conclusion

The aim of this project was to develop an online student exeat system for the Mountain Top University for the purpose of exeat application and management. The project was to implement it with the use of technologies like the Microsoft Visual Studio 2017, Microsoft SQL Server 2014, Entity Framework, C sharp programming language and SQL. The process of exeat application was thus simplified and converted into a web application solution (OSES) and implemented.

References

Dutta, S. (2009). *Global information technology report 2008-2009.*

Frischholz, R. W., & Dieckmann, U. J. C. (2000). *BiolD: a multimodal biometric identification system. 33*(2), 64-68.

Kadir, H. B. A., Wahab, M. H. A., & Kanafiah, S. N. A. B. M. (2009). Boarding School Students Monitoring Systems (E-ID) Using Radio Frequency Identification. *Journal of Social Sciences*, 5(3), 206-211.

KissFlow. *Rapid Application Development Life Cycle*. (2018) Retrieved from https://kissflow.com/rad/rapid-application-development/

Leave Master. (2014). Retrieved from http://www.leavemaster.com/.

Mojares, P. V., Litan, G. A. T., & Mojares, J. G. (2013). INOTIFIED: AN SMS AND RFID-BASED NOTIFICATION SYSTEM OF LIPA CITY COLLEGES, LIPA CITY, BATANGAS, PHILIPPINES. *Journal of Applied Global Research,* 6(18).

Olaniyi, O.M, Omotosho. A, Oluwatosin E.A, Adegoke M.A & Akinmukomi, T (2012). *Students Exeat Monitoring System Using Fingerprint Biometric Authentication and Mobile Short Message Service.* The Don International Journal of ICT and Youth Development (2012). *2(1), 76-85.*

Omidiora, E., Olabiyisi, S., Arulogun, O., Oyeleye, C., & Adegbola, A. J. A. P., Obafemi Awolowo University. Ile-Ife. Nigeria. (2009). *A Prototype of An Access Control System For a Computer Laboratory Scheduling.* 114-120.

Onuiri, E. E., Odukoya, A., Yadeka, C. & Nzei, M (2014). *Design and Implementation of an Electronic Exeat System*. International Journal of Scientific and Engineering Research. 5(4).

Online Leave Management System (2019): Retrieved from http://www.planmyleave.com June 17th 2019.

Rapid Application Development Life Cycle. Retrieved from https://kissflow.com/rad/rapid-application-development/ on the 17th June 2019.

Singhal, Z., & Gujral, R. K. (2012). Anytime anywhere-remote monitoring of attendance system based on RFID using GSM network. *International Journal of Computer Applications*, 39(3), 37-41.

APPENDICES

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;

namespace OSES.Models.OSES
{
    public class Exeat
    {
        public int Id { get; set; }
        public bool official { get; set; }
        public string Title { get; set; }
        public string Body { get; set; }
        public string Depart { get; set; }
        public string Arrive { get; set; }

        public Student user { get; set; }

        public string ExeatType { get; set; }
        public ExeatType Type { get; set; }

        public bool HODStatus { get; set; }
        public bool SAStatus { get; set; }
    }
  }

using System;
using System.Collections.Generic;
using System.ComponentModel.DataAnnotations;
using System.Linq;
using System.Web;

namespace OSES.Models.OSES
{
    public class Student
    {
        [Key]
        public int matric_no { get; set; }
        public string first_name { get; set; }
        public string last_name  { get; set; }
        public string othername { get; set; }
        public int room_no { get; set; }

        public string HallofReside { get; set; }
        public HallofReside HType { get; set; }

        public string Programme { get; set; }
        public Programme Type { get; set; }

        public RegisterViewModel user { get; set; }

        public List<Exeat> Exeat { get; set; }
    }
```

```
}

using System.Data.Entity;
using System.Security.Claims;
using System.Threading.Tasks;
using Microsoft.AspNet.Identity;
using Microsoft.AspNet.Identity.EntityFramework;

namespace OSES.Models
{
    // You can add profile data for the user by adding more properties to your
ApplicationUser class, please visit https://go.microsoft.com/fwlink/?LinkID=317594 to
learn more.
    public class ApplicationUser : IdentityUser
    {
        public async Task<ClaimsIdentity>
GenerateUserIdentityAsync(UserManager<ApplicationUser> manager)
        {
            // Note the authenticationType must match the one defined in
CookieAuthenticationOptions.AuthenticationType
            var userIdentity = await manager.CreateIdentityAsync(this,
DefaultAuthenticationTypes.ApplicationCookie);
            // Add custom user claims here
            return userIdentity;
        }
    }

    public class ApplicationDbContext : IdentityDbContext<ApplicationUser>
    {
        public ApplicationDbContext()
            : base("DefaultConnection", throwIfV1Schema: false)
        {
        }

        public static ApplicationDbContext Create()
        {
            return new ApplicationDbContext();
        }
    }
}

using OSES.Models.OSES;
using System;
using System.Collections.Generic;
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;

namespace OSES.Models
{
    public class ExternalLoginConfirmationViewModel
    {
        [Required]
        [Display(Name = "Email")]
        public string Email { get; set; }
    }

    public class ExternalLoginListViewModel
    {
        public string ReturnUrl { get; set; }
    }

    public class SendCodeViewModel
    {
        public string SelectedProvider { get; set; }
        public ICollection<System.Web.Mvc.SelectListItem> Providers { get; set; }
        public string ReturnUrl { get; set; }
        public bool RememberMe { get; set; }
    }

    public class VerifyCodeViewModel
    {
        [Required]
        public string Provider { get; set; }

        [Required]
        [Display(Name = "Code")]
        public string Code { get; set; }
        public string ReturnUrl { get; set; }

        [Display(Name = "Remember this browser?")]
        public bool RememberBrowser { get; set; }

        public bool RememberMe { get; set; }
    }

    public class ForgotViewModel
    {
        [Required]
        [Display(Name = "Email")]
        public string Email { get; set; }
    }

    public class LoginViewModel
    {
        [Required]
        [Display(Name = "Email")]
        [EmailAddress]
        public string Email { get; set; }

        [Required]
        [DataType(DataType.Password)]
        [Display(Name = "Password")]
        public string Password { get; set; }

        [Display(Name = "Remember me?")]
        public bool RememberMe { get; set; }
    }

    public class RegisterViewModel
    {
        [DatabaseGenerated(DatabaseGeneratedOption.Identity)]
        [Key]
        [Required]
        public string Id { get; set; }

        [Required]
        [EmailAddress]
        [Display(Name = "Email")]
        public string Email { get; set; }

        [Required]
        //[StringLength(100, ErrorMessage = "The {0} must be at least {2} characters
long.", MinimumLength = 6)]
        [DataType(DataType.Password)]
        [Display(Name = "Password")]
        public string Password { get; set; }

        [DataType(DataType.Password)]
        [Display(Name = "Confirm password")]
        [Compare("Password", ErrorMessage = "The password and confirmation password do
not match.")]
        public string ConfirmPassword { get; set; }

        public IList<Student> Students { get; set; }
    }

    public class ResetPasswordViewModel
    {
        [Required]
        [EmailAddress]
        [Display(Name = "Email")]
        public string Email { get; set; }

        [Required]
        //[StringLength(100, ErrorMessage = "The {0} must be at least {2} characters
long.", MinimumLength = 6)]
        [DataType(DataType.Password)]
        [Display(Name = "Password")]
        public string Password { get; set; }

        [DataType(DataType.Password)]
        [Display(Name = "Confirm password")]
        [Compare("Password", ErrorMessage = "The password and confirmation password do
not match.")]
        public string ConfirmPassword { get; set; }

        public string Code { get; set; }
    }

    public class ForgotPasswordViewModel
    {
        [Required]
        [EmailAddress]
        [Display(Name = "Email")]
        public string Email { get; set; }
    }
}
using System.Collections.Generic;
using System.ComponentModel.DataAnnotations;
using Microsoft.AspNet.Identity;
using Microsoft.Owin.Security;

namespace OSES.Models
{
    public class IndexViewModel
    {
        public bool HasPassword { get; set; }
        public IList<UserLoginInfo> Logins { get; set; }
        public string PhoneNumber { get; set; }
        public bool TwoFactor { get; set; }
        public bool BrowserRemembered { get; set; }
    }

    public class ManageLoginsViewModel
    {
        public IList<UserLoginInfo> CurrentLogins { get; set; }
        public IList<AuthenticationDescription> OtherLogins { get; set; }
    }

    public class FactorViewModel
    {
        public string Purpose { get; set; }
    }

    public class SetPasswordViewModel
    {
        [Required]
        //[StringLength(100, ErrorMessage = "The {0} must be at least {2} characters
long.", MinimumLength = 6)]
        [DataType(DataType.Password)]
        [Display(Name = "New password")]
        public string NewPassword { get; set; }

        [DataType(DataType.Password)]
        [Display(Name = "Confirm new password")]
        [Compare("NewPassword", ErrorMessage = "The new password and confirmation
password do not match.")]
        public string ConfirmPassword { get; set; }
    }

    public class ChangePasswordViewModel
    {
        [Required]
        [DataType(DataType.Password)]
        [Display(Name = "Current password")]
        public string OldPassword { get; set; }

        [Required]
        //[StringLength(100, ErrorMessage = "The {0} must be at least {2} characters
long.", MinimumLength = 6)]
        [DataType(DataType.Password)]
        [Display(Name = "New password")]
        public string NewPassword { get; set; }

        [DataType(DataType.Password)]
        [Display(Name = "Confirm new password")]
        [Compare("NewPassword", ErrorMessage = "The new password and confirmation
password do not match.")]
        public string ConfirmPassword { get; set; }
    }

    public class AddPhoneNumberViewModel
    {
        [Required]
        [Phone]
        [Display(Name = "Phone Number")]
        public string Number { get; set; }
    }

    public class VerifyPhoneNumberViewModel
    {
        [Required]
        [Display(Name = "Code")]
        public string Code { get; set; }

        [Required]
        [Phone]
        [Display(Name = "Phone Number")]
        public string PhoneNumber { get; set; }
    }

    public class ConfigureTwoFactorViewModel
    {
        public string SelectedProvider { get; set; }
        public ICollection<System.Web.Mvc.SelectListItem> Providers { get; set; }
    }
}
```

```
<?xml version="1.0" encoding="utf-8"?>
<!--
  For more information on how to configure your ASP.NET application, please visit
  https://go.microsoft.com/fwlink/?LinkId=301880
  -->
<configuration>
  <configSections>
    <!-- For more information on Entity Framework configuration, visit
http://go.microsoft.com/fwlink/?LinkID=237468 -->
    <section name="entityFramework"
type="System.Data.Entity.Internal.ConfigFile.EntityFrameworkSection, EntityFramework,
Version=6.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089"
requirePermission="false" />
  </configSections>
  <connectionStrings>
    <add name="DefaultConnection" connectionString="Data
Source=(LocalDb)\MSSQLLocalDB;AttachDbFilename=|DataDirectory|\OSES.mdf;Initial
Catalog=OSES;Integrated Security=True"
      providerName="System.Data.SqlClient" />
  </connectionStrings>
  <appSettings>
    <add key="webpages:Version" value="3.0.0.0" />
    <add key="webpages:Enabled" value="false" />
    <add key="ClientValidationEnabled" value="true" />
    <add key="UnobtrusiveJavaScriptEnabled" value="true" />
  </appSettings>
  <system.web>
    <authentication mode="None" />
    <compilation debug="true" targetFramework="4.5.2" />
    <httpRuntime targetFramework="4.5.2" />
    <httpModules>
```

```
      <add name="ApplicationInsightsWebTracking"
type="Microsoft.ApplicationInsights.Web.ApplicationInsightsHttpModule, Microsoft.AI.Web"
/>
    </httpModules>
  </system.web>
  <system.webServer>
    <modules>
      <remove name="FormsAuthentication" />
      <remove name="ApplicationInsightsWebTracking" />
      <add name="ApplicationInsightsWebTracking"
type="Microsoft.ApplicationInsights.Web.ApplicationInsightsHttpModule, Microsoft.AI.Web"
preCondition="managedHandler" />
    </modules>
    <validation validateIntegratedModeConfiguration="false" />
  </system.webServer>
  <runtime>
    <assemblyBinding xmlns="urn:schemas-microsoft-com:asm.v1">
      <dependentAssembly>
        <assemblyIdentity name="Microsoft.Owin.Security"
publicKeyToken="31bf3856ad364e35" />
        <bindingRedirect oldVersion="1.0.0.0-3.0.1.0" newVersion="3.0.1.0" />
      </dependentAssembly>
      <dependentAssembly>
        <assemblyIdentity name="Microsoft.Owin.Security.OAuth"
publicKeyToken="31bf3856ad364e35" />
        <bindingRedirect oldVersion="1.0.0.0-3.0.1.0" newVersion="3.0.1.0" />
      </dependentAssembly>
      <dependentAssembly>
        <assemblyIdentity name="Microsoft.Owin.Security.Cookies"
publicKeyToken="31bf3856ad364e35" />
        <bindingRedirect oldVersion="1.0.0.0-3.0.1.0" newVersion="3.0.1.0" />
      </dependentAssembly>
      <dependentAssembly>
        <assemblyIdentity name="Microsoft.Owin" publicKeyToken="31bf3856ad364e35" />
        <bindingRedirect oldVersion="1.0.0.0-3.0.1.0" newVersion="3.0.1.0" />
      </dependentAssembly>
      <dependentAssembly>
        <assemblyIdentity name="Newtonsoft.Json" culture="neutral"
publicKeyToken="30ad4fe6b2a6aeed" />
        <bindingRedirect oldVersion="0.0.0.0-6.0.0.0" newVersion="6.0.0.0" />
      </dependentAssembly>
      <dependentAssembly>
        <assemblyIdentity name="System.Web.Optimization"
publicKeyToken="31bf3856ad364e35" />
        <bindingRedirect oldVersion="1.0.0.0-1.1.0.0" newVersion="1.1.0.0" />
      </dependentAssembly>
      <dependentAssembly>
        <assemblyIdentity name="WebGrease" publicKeyToken="31bf3856ad364e35" />
        <bindingRedirect oldVersion="0.0.0.0-1.5.2.14234" newVersion="1.5.2.14234" />
      </dependentAssembly>
      <dependentAssembly>
        <assemblyIdentity name="System.Web.Helpers" publicKeyToken="31bf3856ad364e35" />
        <bindingRedirect oldVersion="1.0.0.0-3.0.0.0" newVersion="3.0.0.0" />
      </dependentAssembly>
      <dependentAssembly>
        <assemblyIdentity name="System.Web.Mvc" publicKeyToken="31bf3856ad364e35" />
        <bindingRedirect oldVersion="1.0.0.0-5.2.3.0" newVersion="5.2.3.0" />
      </dependentAssembly>
      <dependentAssembly>
        <assemblyIdentity name="System.Web.WebPages" publicKeyToken="31bf3856ad364e35" />
        <bindingRedirect oldVersion="1.0.0.0-3.0.0.0" newVersion="3.0.0.0" />
      </dependentAssembly>
    </assemblyBinding>
  </runtime>
  <entityFramework>
    <defaultConnectionFactory
type="System.Data.Entity.Infrastructure.LocalDbConnectionFactory, EntityFramework">
      <parameters>
        <parameter value="mssqllocaldb" />
      </parameters>
    </defaultConnectionFactory>
    <providers>
      <provider invariantName="System.Data.SqlClient"
type="System.Data.Entity.SqlServer.SqlProviderServices, EntityFramework.SqlServer" />
    </providers>
  </entityFramework>
  <system.codedom>
    <compilers>
      <compiler language="c#;cs;csharp" extension=".cs"
type="Microsoft.CodeDom.Providers.DotNetCompilerPlatform.CSharpCodeProvider,
Microsoft.CodeDom.Providers.DotNetCompilerPlatform, Version=1.0.3.0, Culture=neutral,
PublicKeyToken=31bf3856ad364e35" warningLevel="4" compilerOptions="/langversion:6
/nowarn:1659;1699;1701" />
      <compiler language="vb;vbs;visualbasic;vbscript" extension=".vb"
type="Microsoft.CodeDom.Providers.DotNetCompilerPlatform.VBCodeProvider,
Microsoft.CodeDom.Providers.DotNetCompilerPlatform, Version=1.0.3.0, Culture=neutral,
PublicKeyToken=31bf3856ad364e35" warningLevel="4" compilerOptions="/langversion:14
/nowarn:41008 /define:_MYTYPE=\&quot;Web\&quot; /optionInfer+" />
    </compilers>
  </system.codedom>
</configuration>

<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>@ViewBag.Title - OSES</title>
    @Styles.Render("~/Content/css")
    @Scripts.Render("~/bundles/modernizr")

</head>
<body>
    <div class="navbar navbar-inverse navbar-fixed-top">
        <div class="container">
            <div class="navbar-header">
                <button type="button" class="navbar-toggle" data-toggle="collapse" data-
target=".navbar-collapse">
                    <span class="icon-bar"></span>
                    <span class="icon-bar"></span>
                    <span class="icon-bar"></span>
                </button>
                @Html.ActionLink("OSES", "Index", "Home", new { area = "" }, new { @class
= "navbar-brand" })
            </div>
            <div class="navbar-collapse collapse">
```

```
                <ul class="nav navbar-nav">
                    <li>@Html.ActionLink("Home", "Index", "Home")</li>
                    <li>@Html.ActionLink("About", "About", "Home")</li>
                    <li>@Html.ActionLink("Contact", "Contact", "Home")</li>
                    @if (User.IsInRole("User")|| User.IsInRole("HOD")||
User.IsInRole("VC")|| User.IsInRole("Admin"))
                    {
                        <li>@Html.ActionLink("Exeats", "Index", "Exeats")</li>
                    }
                    @if (User.IsInRole("Admin"))
                    {
                        <li>@Html.ActionLink("Hall of Residence", "Index",
"HallofResides")</li>
                        <li>@Html.ActionLink("Exeat Types", "Index", "ExeatTypes")</li>
                        <li>@Html.ActionLink("Departments", "Index", "Programmes")</li>
                        <li>@Html.ActionLink("Students", "Index", "Students")</li>
                    }

                    @if (User.IsInRole("User"))
                    {
                        <li>@Html.ActionLink("Students", "Index", "Students")</li>
                    }
                </ul>
                @Html.Partial("_LoginPartial")
            </div>
        </div>
    </div>
    <div class="container body-content">
        @RenderBody()
        <hr />
        <footer>
            <p>&copy; @DateTime.Now.Year - Online Student Exeat System</p>
        </footer>
    </div>

    @Scripts.Render("~/bundles/jquery")
    @Scripts.Render("~/bundles/bootstrap")
    @RenderSection("scripts", required: false)
</body>
</html>


@model OSES.Models.OSES.Exeat

@{
    ViewBag.Title = "Create";
}

<h2>Create</h2>


@using (Html.BeginForm())
{
    @Html.AntiForgeryToken()

    <div class="form-horizontal">
        <h4>Exeat</h4>
        <hr />
        @Html.ValidationSummary(true, "", new { @class = "text-danger" })
        <div class="form-group">
            @Html.LabelFor(model => model.official, htmlAttributes: new { @class =
"control-label col-md-2" })
            <div class="col-md-10">
                <div class="checkbox">
                    @Html.EditorFor(model => model.official)
                    @Html.ValidationMessageFor(model => model.official, "", new { @class
= "text-danger" })
                </div>
            </div>
        </div>

        <div class="form-group">
            @Html.LabelFor(model => model.Title, htmlAttributes: new { @class = "control-
label col-md-2" })
            <div class="col-md-10">
                @Html.EditorFor(model => model.Title, new { htmlAttributes = new { @class
= "form-control" } })
                @Html.ValidationMessageFor(model => model.Title, "", new { @class =
"text-danger" })
            </div>
        </div>

        <div class="form-group">
            @Html.LabelFor(model => model.Body, htmlAttributes: new { @class = "control-
label col-md-2" })
            <div class="col-md-10">
                @Html.EditorFor(model => model.Body, new { htmlAttributes = new { @class
= "form-control" } })
                @Html.ValidationMessageFor(model => model.Body, "", new { @class = "text-
danger" })
            </div>
        </div>

        <div class="form-group">
            @Html.LabelFor(model => model.Depart, htmlAttributes: new { @class =
"control-label col-md-2" })
            <div class="col-md-10">
                @Html.EditorFor(model => model.Depart, new { htmlAttributes = new {
@class = "form-control" } })
                @Html.ValidationMessageFor(model => model.Depart, "", new { @class =
"text-danger" })
            </div>
        </div>

        <div class="form-group">
            @Html.LabelFor(model => model.Arrive, htmlAttributes: new { @class =
"control-label col-md-2" })
            <div class="col-md-10">
                @Html.EditorFor(model => model.Arrive, new { htmlAttributes = new {
@class = "form-control" } })
                @Html.ValidationMessageFor(model => model.Arrive, "", new { @class =
"text-danger" })
            </div>
        </div>

        <div class="form-group">
```

```
            @Html.LabelFor(model => model.user.matric_no, htmlAttributes: new { @class =
"control-label col-md-2" })
            <div class="col-md-10">
                @Html.Display("matrics", null, new { htmlAttributes = new { @class =
"form-control" } })
                @Html.ValidationMessageFor(model => model.user, "", new { @class = "text-
danger" })
            </div>
        </div>

        <div class="form-group">
            @Html.LabelFor(model => model.ExeatType, "ExeatType", htmlAttributes: new {
@class = "control-label col-md-2" })
            <div class="col-md-10">
                @Html.DropDownList("ExeatType", null, new { htmlAttributes = new { @class
= "form-control" } })
                @Html.ValidationMessageFor(model => model.ExeatType, "", new { @class =
"text-danger" })
            </div>
        </div>

        <div class="form-group">
            <div class="col-md-offset-2 col-md-10">
                <input type="submit" value="Create" class="btn btn-default" />
            </div>
        </div>
    </div>
}

<div>
    @Html.ActionLink("Back to List", "Index")
</div>

@section Scripts {
    @Scripts.Render("~/bundles/jqueryval")
}
```