THE UNIVERSITY OF THE WEST INDIES

B.Sc. (Engineering)

Department of Electrical and Computer Engineering

ECNG 3020 – SPECIAL PROJECT

**MULTI-PURPOSE HUMAN RESOURCE INFORMATION SYSTEM**

FINAL REPORT

Avron Ramroop

816021311

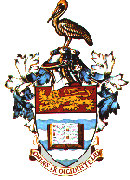
October 16 2008

ELECTRICAL

& COMPUTER

ENGINEERING

DEPARTMENT



Project Supervisor: Dr. Marcus George

Project Type: IV

Date Submitted: 29/03/2023

**Abstract**

Human resource information systems are used to collect and store information related to the employees of an organisation. This information can be used to foster growth in effeciency within the organisation. This project seeks to design and implement a cloud based HRIS to be used to store and manage information of staff members of the Department of Electrical and Computer Engineering at the University of the West Indies, St. Augustine. The waterfall software development model was used to conceptualise the development process so the system was first designed using data flow diagrams before being implemented in code. These system designs were implemented using the Django Framework and was hosted using ‘PythonAnywhere.’

**Table of Contents**

[List of Figures vi](#_Toc130980411)

[List of Tables vii](#_Toc130980412)

[List of Abbreviations viii](#_Toc130980413)

[1. Introduction 10](#_Toc130980414)

[1.1 Background 10](#_Toc130980415)

[1.2 Motivation 10](#_Toc130980416)

[1.3 Aim and Objectives 11](#_Toc130980417)

[1.4 Scope 13](#_Toc130980418)

[1.5 Outline 13](#_Toc130980419)

[2. Literature Review 14](#_Toc130980420)

[2.1 Human Resource Record Keeping Standards 14](#_Toc130980422)

[2.2 Previous Work 17](#_Toc130980423)

[3. Methodology 23](#_Toc130980424)

[3.1 Software Development Model 23](#_Toc130980426)

[3.1.1 The Waterfall Model 23](#_Toc130980427)

[3.1.2 Prototyping Development Model 24](#_Toc130980428)

[3.1.3 Spiral Development Model 25](#_Toc130980429)

[3.1.4 Unified Development Model 27](#_Toc130980430)

[3.1.5 Comparison of Software Development Models 28](#_Toc130980431)

[3.2 Requirements Definition 30](#_Toc130980432)

[3.3 Software and Framework 31](#_Toc130980433)

[3.4 System and Software Design 33](#_Toc130980434)

[3.4.1 System Architecture 33](#_Toc130980435)

[3.4.2 Data Design 36](#_Toc130980436)

[3.4.3 Data Flow Diagrams 39](#_Toc130980437)

[3.4.4 User Interface Design 42](#_Toc130980438)

[3.5 Design Implementation 42](#_Toc130980439)

[3.5.1 HRIS\_APP 43](#_Toc130980440)

[3.5.2 STAFF\_APP 46](#_Toc130980441)

[3.5.3 ADMIN\_APP 50](#_Toc130980442)

[3.5.4 User Interface 52](#_Toc130980443)

[3.5.5 Other Functions 52](#_Toc130980444)

[3.5.6 Hosting 53](#_Toc130980445)

[4. Results 54](#_Toc130980446)

[4.1 User Interface 54](#_Toc130980448)

[4.1.1 Staff Pages 55](#_Toc130980449)

[4.1.2 Admin Pages 57](#_Toc130980450)

[4.2 Test Results 61](#_Toc130980451)

[4.2.1 HRIS\_APP 61](#_Toc130980452)

[4.2.2 STAFF\_APP 65](#_Toc130980453)

[4.2.3 ADMIN\_APP 67](#_Toc130980454)

[5. Conclusion 69](#_Toc130980455)

[References 70](#_Toc130980456)

[Appendix A: Software Requirements Specification 72](#_Toc130980457)

[Appendix B: Software Design Description 84](#_Toc130980477)

[Appendix C: Code Snippets 110](#_Toc130980500)

# List of Figures

[Figure 1: The Waterfall Model 24](#_Toc130971564)

[Figure 2: Prototyping Development Model 25](#_Toc130971565)

[Figure 3: Spiral Development Model 26](#_Toc130971566)

[Figure 4: Unified Development Model 28](#_Toc130971567)

[Figure 5: High Level Product Perspective Diagram 30](#_Toc130971568)

[Figure 6: System Architecture 33](#_Toc130971569)

[Figure 7: Authentication Module DFD Level 1 40](#_Toc130971570)

[Figure 8: Authentication Module DFD Level 2 40](#_Toc130971571)

[Figure 9: Admin Module DFD Level 2 41](#_Toc130971572)

[Figure 10: Staff Module DFD Level 2 41](#_Toc130971573)

[Figure 11: Default Django App Files 43](#_Toc130971574)

[Figure 12: ER Diagram for Database Models 47](#_Toc130971575)

[Figure 13: Login Page 54](#_Toc130971576)

[Figure 14: Registration Page 54](#_Toc130971577)

[Figure 15: Staff Home Page 55](#_Toc130971578)

[Figure 16: Staff Profile Page 55](#_Toc130971579)

[Figure 17: Staff Profile Page Continued 56](#_Toc130971580)

[Figure 18: Staff View Contacts 56](#_Toc130971581)

[Figure 19: Staff Submit Request For Information Page 57](#_Toc130971582)

[Figure 20: Admin Home Page 57](#_Toc130971583)

[Figure 21: Admin View Employees Page 58](#_Toc130971584)

[Figure 22: Admin Add Account Page 58](#_Toc130971585)

[Figure 23: Admin Batch Add Accounts Page 59](#_Toc130971586)

[Figure 24: Admin View Contacts Page 59](#_Toc130971587)

[Figure 25: Admin Annual Reports Page 1/3 60](#_Toc130971588)

[Figure 26: Admin Annual Reports Page 2/3 60](#_Toc130971589)

[Figure 27: Admin Annual Reports Page 3/3 61](#_Toc130971590)

# List of Tables

[Table 1: Comparison of 4 Different Software Development Models 28](#_Toc130980281)

[Table 2: Admin Sub Module Descriptions 34](#_Toc130980282)

[Table 3: Staff Sub Modules Descriptions 35](#_Toc130980283)

[Table 4: User Information 36](#_Toc130980284)

[Table 5: Employee Information 36](#_Toc130980285)

[Table 6:Publications 37](#_Toc130980286)

[Table 7: Awards 37](#_Toc130980287)

[Table 8: Unpublished Manuscripts 37](#_Toc130980288)

[Table 9: Honours and Certificates 37](#_Toc130980289)

[Table 10: Areas of Academic Specialisation 37](#_Toc130980290)

[Table 11: Research Grants 37](#_Toc130980291)

[Table 12: Post Graduate Research and Supervision 37](#_Toc130980292)

[Table 13: Research Interests and Associations 38](#_Toc130980293)

[Table 14: Conferences Attended 38](#_Toc130980294)

[Table 15: Technical Presentations 38](#_Toc130980295)

[Table 16: University Professional Service Roles 38](#_Toc130980296)

[Table 17: Professional Consultancies 38](#_Toc130980297)

[Table 18: Professional Development 39](#_Toc130980298)

[Table 19: Contributions to the Department, Faculty or University 39](#_Toc130980299)

[Table 20: Other Professional Activities 39](#_Toc130980300)

[Table 21: Request to the Administrator 39](#_Toc130980301)

[Table 22: Contacts 39](#_Toc130980302)

[Table 23: Registration Function Test Results 62](#_Toc130980303)

[Table 24: Login Function Test Results 62](#_Toc130980304)

[Table 25: Add Account Function Test Results 63](#_Toc130980305)

[Table 26: Batch Add Function Test Results 64](#_Toc130980306)

[Table 27: View Model Data Functions Test Results 65](#_Toc130980307)

[Table 28: Add Model Object Functions Test Results 65](#_Toc130980308)

[Table 29: Edit Model Object Functions Test Results 66](#_Toc130980309)

[Table 30: Delete Model Object Functions Test Results 67](#_Toc130980310)

[Table 31: View All Employees Function Test Results 67](#_Toc130980311)

[Table 32: View Employee Function Test Results 68](#_Toc130980312)

[Table 33: Annual Reports Function Test Results 68](#_Toc130980313)

# List of Abbreviations

AI – Artificial Intelligence

B/S – Browser/Server

CSS – Cascading Style Sheets

DFD – Data Flow Diagram

ER – Entity Relationship

GDPR – General Data Protection Regulation

HIPPAA – Health Insurance Portability and Accountability Act

HRIS – Human Resource Information System

HRM – Human Resource Management

HTML – Hypertext Markup Language

HTTPS – Hypertext Transfer Protocol Secure

IEEE – Institute of Electrical and Electronics Engineers

ILO – International Labour Organisation

Int – Integer Field

ISO – International Organisation for Standardisation

IT- Information Technology

ML – Machine Learning

MVC – Model-View-Controller

ORM – Object Relational Mapping

PHP – Hypertext Pre-processor

PDF – Portable Document Format

SDD – Software Design Description

SMTP – Simple Mail Transfer Protocol

SOX – Sarbanes-Oxley Act

SQL – Structured Query Language

SRS – Software Requirements Specification

UDM – Unified Development Model

UI – User Interface

URL – Uniform Resource Locator

Varchar – Variable Character Field

# Introduction

## Background

Human Resource Information Systems (HRIS) are the “integration of software, hardware, support functions, and system policies and procedures into an automated process designed to support the strategic and operational activities of the human resources department and managers throughout the organization” (Chauhan, Sharma and Tyagi 2011). These systems are a combination of information technology (IT) and human resource management (HRM). They contain information about employees to be used for training and motivation purposes by the administration of the organisation. The collection of employee data to be utilised in this manner fosters efficient decision making by the administration of the organisation. This is the main goal of these systems. (Barisic, Tomic and Bach 2022).

## Motivation

The Department of Electrical and Computer Engineering at the University of the West Indies, St. Augustine currently does not have a singular system to manage the human resource information of all its staff. Most information is spread across various software so the creation of a singular point to manage and stored this host of scattered information with help to streamline the administrative department’s operation and aid in overall productivity of the faculty as a whole.

## Aim and Objectives

This project is aimed towards developing a cloud based human resource management system to be used by the administrative staff of the department of Electrical and Computer Engineering. The specific objectives are listed below:

1. Perform a literature survey on existing Human Resource Information Systems along with a review of all relevant human resource record keeping standards for both Trinidad and Tobago, regionally and internationally.
2. Create a Software Requirements Specification (SRS) for development of the Multi-Purpose Human Resource Information System.
3. Create a Software Design Description (SDD) for development of the Multi-Purpose Human Resource Information System.
4. Design the Multi-Purpose Human Resource Information System with the following features.
   1. Allows administrator to create or reset HRIS accounts, both manually and through a batch process.
   2. Allow staff to modify password after creation or reset of HRIS accounts.
   3. Sends all staff members Request for Information (RFI) alerts in the in the form of emails to stored email address. The nature of the requested information must be indicated in the email.
   4. Allows staff to directly update essential data to the system. The staff will be responsible for updating their profile. The essential data is as follows:
      1. Bio data including name, date of birth, contact number, email address, NIS number, PAYE number, etc.
      2. Updated Curriculum Vitae in .DOCX or PDF
      3. Publication Achievements with academic year - Journal papers, Conference Papers, Books
      4. Awards within academic year
      5. Fellowships, Memberships, Affiliations with academic year
   5. Allows managers to generate a variety of reports based on data entered by staff, e.g. Annual Reports consisting of staff publication list, awards, fellowships, even documents required for staff evaluation, leave management, etc.
   6. Allows staff to request information from the system via request to the administrator, which will be displayed for viewing purposes. Information to be requested are as follows:
      1. Contract benefits including basic salary, allowances and all book and study grants entitled to.
      2. Staff training opportunities including short courses, certificate programmes, etc.
      3. List of all important university contacts for important operations such as applying for study leave, submission of book grant claims, etc.
5. Use an appropriate software package to implement the Multi-Purpose Human Resource Information System designed in objective 4.
6. Outline ten (10) unique sets of test cases required for verification of each feature of the Multi-Purpose Human Resource Information System implemented in objective 5.
7. Test the Multi-Purpose Human Resource Information System using test cases of objective 6.

## Scope

This project seeks to design and implement a cloud based Human Resource Information System to be used by the administrative staff of the Electrical and Computer Engineering Department to store and organise the human resource information of the staff. The administrative staff are therefore the stakeholders of the system to be created and will be referred to as such for the remainder of this document.

## Outline

This report is arranged in a linear fashion. Section one involves the background information and reason for undertaking the project. Section two explores the past work in the field of HRIS and the global standards of human resource record keeping. Section three discusses the methodology behind the designs of the system as well as the approach to software engineering taken. Section four displays the results achieved from implementing the completed system with an appropriate software package and the final section concludes the report and outlines whether the objectives were achieved.

# Literature Review



## Human Resource Record Keeping Standards

The act of recording human resource information, whether that be electronically or physically in hard copy format, must be performed with adherence to certain standards. These standards generally apply to any institution which stores the information associated with each of its employees. Human resource record keeping standards vary by country, but there are common themes that are maintained throughout from country to country. These common practices are outlined below:

* Personal information: Businesses are required to maintain correct and current records of all personal information relating to their employees, including their complete name, contact information, date of birth, and employment status.
* Employment history: Companies are required to maintain records of each employee's employment history, including the start date, job title, and any modifications to that status.
* Payroll and benefits: Companies are required to keep track of all payments made to employees, including salaries, bonuses, benefits, and taxes deducted.
* Employee performance and disciplinary actions: Companies must maintain records of each employee's performance as well as any disciplinary actions that may have been taken.
* Safety and health: Companies are required to maintain records of workplace accidents, illnesses, and audits, training programs, and workplace inspections.
* Termination: Companies are required to maintain records regarding the termination of employees, including those pertaining to severance payments, redundancy packages, and exit interviews.

Internationally there are several laws and regulations as well as standards geared towards the recording, management and security of personal information related to the employees of an organisation. The most pertinent and relevant standards and regulations are shown below:

* General Data Protection Regulation (GDPR): This European Union (EU) law establishes standards for the gathering, handling, and archiving of personal data. Regardless of where the organization is situated, it applies to all organizations that gather data on EU citizens.
* ISO 27001 is a widely accepted international standard for information security management systems. It was developed by the International Organization for Standardization (ISO). It offers a framework for setting up and maintaining security measures to safeguard information assets, including personnel records.
* The Sarbanes-Oxley Act (SOX) is a federal law in the United States that mandates effective internal controls over financial reporting be established and maintained by publicly traded firms. It contains clauses that deal with the storage and defence of employee data.
* The Health Insurance Portability and Accountability Act (HIPAA) is a US federal statute that establishes guidelines for the security of private medical data. It establishes rules for the storage and protection of employee health-related data and is applicable to employers who offer health insurance to their workers.
* Conventions of the International Labour Organization (ILO): International labour standards are developed and promoted by the ILO, a specialized United Nations agency. Some of its treaties, including Convention No. 108 on the Protection of Personal Data and Convention No. 95 on the Preservation of Wages, deal with the protection of employee data.

Employers may also follow other relevant national and international standards, regulations, and best practices for the collection, processing, storage, and protection of human resource information, depending on their industry and location.

In Trinidad and Tobago, the knowledge and consent of an employee is necessary when collecting or disclosing personal information. There are various laws and regulations that govern the standards for human resource record keeping. These include:

* The Employment (Records) Act specifies the obligations of employers to maintain records of their employees' personal information, employment history, and compensation. Additionally, it mandates that companies maintain records of all work visas and immigration papers for foreign employees.
* The Occupational Safety and Health Act mandates that businesses maintain records of illnesses and injuries that occur on the job, as well as records of workplace audits, inspections, and training initiatives.
* The Income Tax Act: This law mandates that businesses maintain records pertaining to tax payments and deductions, including those pertaining to employee salaries, wages, and bonuses.
* The National Insurance Act: This law mandates the keeping of social security records by companies, including records of employee contributions and benefits.
* The Retrenchment and Severance Benefits Act: This law mandates that companies maintain documents about employee termination, including records of severance payments and redundancy benefits.

Overall, Trinidad and Tobago's standards for human resource record keeping stress the significance of keeping accurate and thorough records pertaining to employee information, safety, taxes, and termination. Employers are responsible for making sure that these records are maintained private, secure, and accessible for inspection by the appropriate authorities.

In general, human resource record keeping standards are meant to make sure that companies retain accurate and thorough records of information about employees, their safety, pay, performance, and termination. This aids in safeguarding workers' rights, ensuring adherence to employment laws and regulations, and providing a transparent account of an organization's human resources activities.

## Previous Work

M. Kumar et al 2022 investigated the use of machine learning solutions (ML) and artificial intelligence (AI) based models to increase the accuracy and efficiency of HRM operational processes. The research identified training models for three main domains in the field of HRM where ML can be employed. These domains were employee engagement, culture, and appraisal systems. The main one of interest was employee engagement and for this domain, machine learning models were used to track information about the employees in real time such as issues that they may have. Studying this information could lead to robust prediction models that are able to predict and deal with these issues in an efficient manner thus breed a better workplace environment where employees are more likely to produce high quality work. (Kumar, et al. 2022)

F. Tian 2022 explored the use of the Apriori Algorithm to increase the efficiency and accuracy of HRM systems. The Apriori Algorithm itself is an association-based algorithm which aims to identify underlying relations between different items. It is a model that continuously determines the number of transactions containing two or more subsequent items as compared to transactions with one the singular items. Continuously keeping track of these relationships allows for helpful associations to be made within the database of human resource information. The HRIS itself was divided into 5 modules and extensive testing of the system was performed whereby each module was used to perform allocation of specific data into grouped categories and subcategories according to the determination of the Apriori Algorithm. The results of this testing showed each module with over 90% operation capacity and the system was able to meet the service requirements. (Tian 2022)

Peng 2022 examined the use of fuzzy data mining practices in the design and implementation of a human resource decision making system. Fuzzy data mining is a type of data mining that deals with data that is uncertain, imprecise, or incomplete. It is a method of extracting useful information and patterns from data that is not clearly defined or structured, using techniques from the field of fuzzy logic. This fuzzy data mining approach is used in conjunction with the ID3 algorithm to store the data into a decision tree structure. The ID3 algorithm works by iteratively dividing the data into smaller and smaller subsets based on the features or attributes of the data. At each step, the algorithm selects the feature that best splits the data into two subsets, based on a measure of information gain. The process continues until the data is fully partitioned, or until some stopping criteria are met. Relating and storing the data in this way allowed for an increase in effeciency over traditional HRM systems of 7.2%. (Peng 2022)

Luo 2021 investigated the use of the model of information-based human resource management to design and implement an HRIS based on the B/S architecture, Model-View-Controller (MVC) framework. The B/S architecture is a browser/server structure where a web browser is used as the main front end for an application while most of the computing and transaction logic is implemented on the server side. The MVC framework essentially a software design approach whereby the functionality of your designed application is divided across three main modules: the model, view, and controller. The model component typically handles data, logic and interacts with the database. The controller module is responsible for the communication with the end user as well as the model and view modules to request data and send responses to and from each module. The view module is responsible for producing the interface displayed to the user. Analysis of general human resource management (HRM) theories showed that 5 main functional modules required for successful HRM are recruitment management before joining, file management after joining, behaviour management, training management and growth channel. These main modules were used as a guide to create the fields for the database design. (Mou and Luo 2021)

Zhang et al. 2021 discussed a file and data management system using the open-source Struts-Hibernate Integrated architecture which is a framework based on the MVC framework. The described system utilised this architecture as it allowed for the decoupling of the system into independent modules to improve the flexibility and maintainability of the system. Hibernate was also employed as it uses an object/relational mapping (ORM) system for the files and information, and it also allows for the separation of the system into modules linked to each of the ORM elements. The elements of data themselves were stored in a database created using Access 2010. The combination of these technologies allowed for the fabrication of a robust system to store and access data efficiently. While this specific implementation was used for storing information for a shipbuilding enterprise, the design was flexible enough to be utilised in other fields such as the creation of an HRIS. (Zhang, et al. 2021)

The B/S model was also utilised to implement an HRIS using Python and tree node mining in Ma 2021. The tree node structure is a hierarchical structure that consists of nodes connected by edges. The node at the top of the tree is called the root, and nodes without children are called leaf nodes. Internal nodes are nodes that are not the root or leaf nodes. The relationship between nodes is hierarchical, with a parent node being a node at a higher level and a child node being a node at a lower level. Sibling nodes are nodes that have the same parent and are at the same level in the tree. The depth of a node is the number of edges from the root to the node, and the height of a node is the number of edges in the longest path from the node to a leaf node. An infinite tree is a tree that does not have a root node. This data management structure was utilised by assigning modules of the HRIS to nodes of the tree in such a way that branches link the corresponding subsystems together. The general linking of the subsystems however was said to be unique to the establishment that the HRIS was being constructed for. This approach allows for sharing and extraction of information in a highly efficient manner. (Ma 2021)

Liu 2021 proposed the use of a decision tree algorithm within a human resource management information system. A tree structure was used to represent the decision set of the data. The internal nodes of the tree represent the attribute test while the outputs of the tests are represented by the branches of the tree. The classes are represented by the leaf nodes. The algorithm uses the information entropy of data to assign the root node. The information gain of each set of data is calculated and the attribute with the largest information gain is selected as the root. The child nodes are each assigned on this same basis of information gain using a recursive function until all data attributes are assigned within the decision tree. The use of this type of algorithm on two unique data sets showed an average improvement of around 10% with respect to the accurate classification of information using this algorithm as opposed to traditional methods of data allocation for human resource information systems. (L. Liu 2021)

A smart human resource management system was also created and tested in H. H.A.S.S 2020 using ML and blockchin technology. The constructed system was a web based platform consisting of four main subsystems to assist in the managing of human resource information. These subsystems are the skill assessment module, employee detail verification module,performance predicting module and the attrition predicting module. The main module of interest was the performance predicting model which tested 3 different ML algorithms; Decision tree, Random Forest and Naïve Bayes. The algorithm determined to produce the most accurate results was the Naïve Bayes Gaussian Algorithm. This algorithm is based on applying Bayes' theorem, which is a way to calculate probabilities based on known values. It assumes that the features (or characteristics) of an example are independent of each other, given the class. This is known as the "naive" assumption. It estimates the probability of an example belonging to each class using the probabilities of the features given each class, as well as the overall probability of each class. It predicts the class with the highest probability as the final prediction for the example. The developed prediction model showed an accuracy of 98.7% when used on a given data set and thus this model can be utilised to assist the administrative staff of an organisation in making important decisions about the employees where the overall wellbeing of the establishment is concerned. (H.A.S.S, et al. 2020)

# Methodology

A preliminary review of existing HRIS systems was performed by examining several conference papers published by the Institute of Electrical and Electronics Engineers (IEEE) discussing related work in the field. Using this information along with the outlined objectives of the project, a Software Requirement Specification (SRS) was created to document the design requirements of the project. This can be found in Appendix A.



## Software Development Model

There exist several methods of representing the process by which a software is conceptualised and created. These are known as software process models. They present a high-level abstract model showing the individual process activities but the emphasis on these activities and the order in which they are implemented vary between models (Sommerville 2019). Four development models are examined in the Sections 3.1.1-3.1.5 to determine which is most appropriate approach for conceptualising and implementing the HRIS.

### The Waterfall Model

The Waterfall Model is also known as the linear sequential model as it provides a sequential approach to software development. These steps begin with the consumer as the full requirements of the system are specified. The project then progresses through a full system design followed by implementation of components then testing and finally the operation of the system where it is put to practical use. Each step in the waterfall model is only commenced after the preceding step has been completed. The waterfall model is the simplest to understand and is easy to manage. (Sommerville 2019)

This method however has certain drawbacks. Testing is performed relatively late in the cycle, and this is not ideal as it may lead to errors and delays in the process meaning that this method overall may be regarded as slow oo inefficient. It may also be difficult to implement changes due to the sequential nature of this model. This can pose a problem if the requirements of the system are changed at a later stage in the cycle. (Royce 1970)



Figure 1: The Waterfall Model

### Prototyping Development Model

This model is most utilised in cases where the requirements of the system are not particularly specific or well defined. The Prototyping Development model involves the entire cycle existing as iterations. Each iteration involves several steps geared towards creating a working prototype of the system or of a component of the system allowing for rapid development and easy modifications. (Sommerville 2019)

One of the benefits of this model is that it allows for early identification of issues, such as missing requirements, design flaws, or technical problems, before significant time and resources are invested in the development process. This approach can also help to reduce the overall development time and costs by identifying and addressing potential problems early on.

However, there are some potential drawbacks to the prototyping software development model. For example, it may be difficult to manage the development process when multiple prototypes are being developed simultaneously, and there may be a risk of scope creep if stakeholders request additional features during the development process. Additionally, the prototypes may not be suitable for production use, which can lead to additional development efforts to create a fully functional system. (Pressman and Maxim 2020)



Figure 2: Prototyping Development Model

### Spiral Development Model

The Spiral Development Model is another iterative approach to software development. Each iteration is divided into five main sectors. These sectors are planning, modelling, construction, deployment, and communication. The planning phase of this model involves risk assessment and thus this model is most utilised in complex systems that may involve a high degree of risk factors which may cause delays in production or hinder the overall development of the system in some way. (Boehm 1988)

The main benefit of this model is its flexibility and its accommodation of constant feedback and communication from the consumer or stakeholder at each iteration. This constant collaboration along with the analysis of risk at the early stages of production allow for low likelihood of project failure. (Pressman and Maxim 2020)

However, there are also some potential drawbacks to the spiral model. The risk analysis and evaluation phases can be time-consuming and may require significant resources. Additionally, the model can be complex and may be difficult to implement for small or simple projects. (Pressman and Maxim 2020)

Diagram, engineering drawing

Description automatically generated

Figure 3: Spiral Development Model

*Source:* Pressman and Maxim, 2020.

### Unified Development Model

The Unified Development Model (UDM) is based on the Unified Modelling Language (UML) and specialises in the development of object-oriented software systems. (Jacobson, Booch and Rumbaugh 1999) It is another iterative method that is also incremental and thus provides the “evolutionary feel that is essential in modern software development.” (Pressman and Maxim 2020) UML presents a standardised way of visualising the design of a system and allows clear communication and understanding between multiple developers working on the same system. It also allows complex systems to be simplified and thus allows non-programmers to understand the overall functionality and design of a software system. (GeeksforGeeks 2022)

The main stages of this iterative process are shown in Figure 4. The inception phase focuses on defining the scope of the project and solidifies the understanding of the requirements and constraints between the developers and the stakeholders. The elaboration phase focuses on defining the architecture of the system and modelling the design. The construction phase involves the actual implementation of the designed system complete with testing to ensure that the system is fully functional. The transition phase focuses on deploying and system and maintaining it. (Pressman and Maxim 2020)

The UDM has the benefit of allowing flexibility and emphasises the importance of quality and testing continuously throughout the developmental process. The model however may be too complex and difficult to implement for small projects and may be unnecessary in instances where the developmental team is very small. (Pressman and Maxim 2020)

Diagram

Description automatically generated

Figure 4: Unified Development Model

*Source:* Pressman and Maxim, 2020.

### Comparison of Software Development Models

The main advantages and disadvantages of the 4 software development models analysed are show in Table 1 to summarise the findings and determine which model would be most appropriate for this project.

Table 1: Comparison of 4 Different Software Development Models

|  |  |  |
| --- | --- | --- |
| **Model** | **Advantages** | **Disadvantages** |
| Waterfall | * It is straightforward to grasp and plan. * It functions for simple projects that are clear-cut. * The processes of analysis and testing are simple. | * It struggles to adapt to change. * Testing is at a late stage. * Consumer approval comes last. |
| Prototyping | * The effects of requirement changes are lessened. * The client is involved frequently and early. * It's effective for little projects. * There is a lower chance of product rejection. | * The presence of customers could lead to delays. * The temptation to "ship" a prototype might exist. * A useless prototype wastes work. * It is challenging to manage and plan. |
| Spiral | * The involvement of the consumer is ongoing. * Construction risk is controlled. * Large, sophisticated projects can use it. * It works well for expandable projects | * Project failure due to a flawed risk analysis. * Managing the project could be challenging. * A skilled development team is needed. |
| Unified | * Emphasis is placed on good documentation. * The involvement of the consumer is ongoing. * It adapts to shifting demand. * For maintenance projects, it works nicely. | * Sometimes use cases are not exact. * Integration of the software increments is complicated. * Phase overlaps might be problematic. * A skilled development team is needed. |

*Source:* Pressman and Maxim, 2020.

The iterative approach of the unified, spiral, and prototyping models is favourable in modern software development however the unified model is not suited to small scale systems such as the proposed HRIS system. The spiral model poses management challenges and the level of risk analysis required for this specific system does not warrant its use. The prototyping model is unsuitable as the requirements of this project are very straight forward with little room for changes and this model poses management complications as well. Thus, the waterfall model is chosen as it is the most suitable for smaller scale systems with a low number of developers (one in this case since this is an individual project). The requirements of the proposed HRIS are well defined and robust with little need for revision or change and thus the development of this system would benefit from a simpler model such as the waterfall model. Due to the nature of this project, a modified version of this model will be utilised whereby the implementation and integration will be performed simultaneously as one stage followed by a full system testing stage.

## Requirements Definition

The requirements of the system outlined in section 3.3 item 4 of this document were revised with the relevant stakeholders to solidify the obligations of the system. On a high level the system is required to store information on a cloud-based database. Thus, this warranted a client server architecture where users will access the data through a web-based user interface (UI). The system will be accessed by two user types as seen in Figure 5. The users communicate with the UI via the internet. The UI allows the users to access the functions of the system and thus query information from the database without having to understand the computer code required to communicate with these systems. The HRIS system itself conveys the requests and queries sent through the UI and accesses the database to read/write the required data. The complete requirements of the HRIS system were documented at length and in detail in a Software Requirements Specification (SRS) document which can be found in Appendix A.



Figure 5: High Level Product Perspective Diagram

## Software and Framework

Web-based client/server systems like the system shown in Figure 5 can be decomposed into two main components, the front-end and the back-end. Front-end web development is the process of designing and developing the user interface (UI) of a website or web application. This involves creating the layout, visual design, and interactive elements of the website that users interact with. Back-end web development as the process of designing and developing the server-side logic of a website or web application. This includes designing and managing the database, developing the server-side scripts and APIs, and integrating different systems and services that the website or web application depends on. (Liu, et al. 2017)

The front-end for web-based applications is typically implemented using several different languages such as HTML, CSS, and JavaScript. The UI can be created using any one of these languages separately however they can also be used together to enhance the styling and functionality of the UI. There also exists several front-end development frameworks and libraires utilising these languages such AngularJS, ReactJS and jQuery which are primarily based on JavaScript and incorporate elements of HTML and CSS. (Singhal 2023)

The back-end for web-based applications typically employ using languages such as PHP, C++,

Java or Python. The backend language choice depends on the specific needs of the system to be implemented however for smaller scale applications the choice of language is not very significant to achieving the requirements as the core functionalities of these languages and what they can achieve is relatively the same. (Singhal 2023)

Python was chosen as the programming language to use for the backend of the system as it is a high-level language that is relatively fast and simplistic to use. It was the most used backend programming language of 2022, and its popularity means that it is very well documented and thus it would be easy to find resources to help troubleshoot any issues that arise during development. Python is also regarded as one of the best languages to use when it comes to data analysis and thus it will excel at producing reports of the human resource information. (Dyachenko 2022)

The Django framework was chosen to implement the required HRIS system. The Django framework is based in Python and is an open- source framework with extensive documentation and can be used for development of both the front-end and back-end of a web-based application. It also incorporates the use of other languages such as HTML, CSS, and JavaScript to create the UI of the application. It is a robust system that offers easy scalability allowing for an easy expansion of the project, if necessary, in the future. (Korsun 2022) Django also offers a host of security features. The framework has built in protection against many common security risks such as cross site scripting, cross site request forgery, SQL injections and clickjacking. It also offers deployment via HTTPS protocol. The Django framework also comes with an SQLite database preinstalled which can manage thousands of simultaneous read/write requests and thus it is suitable for application in this system. (D. S. Foundation 2022)

## System and Software Design

### System Architecture

The main system of the HRIS is divided into three main modules for its two types of users. These are the admin module and the staff module and the authentication module. Each of these modules contains their own sub modules as shown in this Figure 6.



Figure 6: System Architecture

Authentication module

The authentication module handles the login and logout of users. It also allows users to change their password if so desired. Upon verification, the user is given access to either the staff module or the admin module depending on the type of the user.

Admin Module

The admin module serves as the main module to be used by the administrator user. It is comprised of several sub modules that these users have access to.

Table 2: Admin Sub Module Descriptions

|  |  |
| --- | --- |
| **Sub Module** | **Description** |
| Employees Sub Module | This module allows admin users to view all employees and all the information stored in the database associated with each employee. This information includes:   * Title * First Name * Middle Name * Last Name * Gender * Phone Number * Bio * Date of Birth * Employee type * Curriculum Vitae * Publications * Awards * Conferences Attended * Professional Development * Manuscripts under review and in preparation * Technical Presentations * Research Grants * Professional Consultancies * University Public and Professional Service Roles * Post Graduate Research and Supervision * General Areas of Academic Specialization * Honours and Certificates obtained * Contribution to Department, Faculty and University * Other Professional Activities |
| Reports Sub Module | The reports submodule of the admin module allows admin users to generate reports of the publications, awards, technical presentations, conferences attended, honours and contributions to the department and university by a specific employee based on the academic year of the aforementioned achievements. The admin user can then generate a pdf file documenting these results. |
| Manage Accounts Sub Module | This sub module allows admin users to view all staff accounts and reset or delete them. The admin user can also add new accounts to the system. |
| Contacts Sub Module | This sub module allows the admin user to edit the page of important university contacts. The user can add or remove contacts to be viewed by the staff user. |
| Requests Sub Module | The requests submodule of the admin module allows admin users to view requests for information submitted by staff users. |

Staff Module

The staff module serves as the main module to be used by the staff user. It is comprised of several sub modules that these users have access to.

Table 3: Staff Sub Modules Descriptions

|  |  |
| --- | --- |
| **Sub Module** | **Description** |
| Profile Sub Module | The profile sub module allows staff users to view and edit their individual profile information. (Information outlined in the employees sub module of Table 2) |
| Requests Sub Module | The requests sub module of the staff module allows a staff user to submit a request for information to be reviewed by the admin user. |
| Contacts Sub Module | The contacts sub module of the staff module allows staff user to view the contact information supplied by the admin users. |

### Data Design

In order to store and manage the data required by the system, several database objects are required. The database information to be stored with their respective data types is outlined below.

Table 4: User Information

|  |  |  |
| --- | --- | --- |
| Column Name | Data Type | Description |
| username | Varchar | Email address |
| password | Varchar | Password of the user |
| user\_type | Varchar | Admin or Staff |

Table 5: Employee Information

|  |  |  |
| --- | --- | --- |
| Column Name | Data Type | Description |
| firstname | Varchar | First Name |
| lastname | Varchar | Surname |
| othername | Varchar | Other Name |
| sex | Varchar | Gender |
| email | Varchar | Email |
| tel | Int | Phone Number |
| bio | Varchar | Biography |
| employeetype | Varchar | Fulltime, Part-time, Contract, Intern |
| nisnumber | Int | NIS Number |
| birthday | date | Date of Birth |
| employeeid | Int | Employee ID Number |
| vitae | file | Curriculum Vitae Document |
| image | file | Profile Image |

Table 6:Publications

|  |  |  |
| --- | --- | --- |
| Column Name | Data Type | Description |
| title | Varchar | Title of Publication |
| publicaitontype | Varchar | Peer Reviewed Journal, Book, Conference Paper |
| year | Varchar | Academic Year |

Table 7: Awards

|  |  |  |
| --- | --- | --- |
| Column Name | Data Type | Description |
| title | Varchar | Title of Award |
| year | Varchar | Academic Year |

Table 8: Unpublished Manuscripts

|  |  |  |
| --- | --- | --- |
| Column Name | Data Type | Description |
| title | Varchar | Title of Manuscript |
| status | Varchar | In Preparation or Under review |

Table 9: Honours and Certificates

|  |  |  |
| --- | --- | --- |
| Column Name | Data Type | Description |
| title | Varchar | Title of Honour/Certificate |
| competition | Varchar | Competition honour was obtained |
| year | Varchar | Academic Year |

Table 10: Areas of Academic Specialisation

|  |  |  |
| --- | --- | --- |
| Column Name | Data Type | Description |
| area | Varchar | Area of Specialisation |

Table 11: Research Grants

|  |  |  |
| --- | --- | --- |
| Column Name | Data Type | Description |
| title | Varchar | Title of Research Grant |

Table 12: Post Graduate Research and Supervision

|  |  |  |
| --- | --- | --- |
| Column Name | Data Type | Description |
| title | Varchar | Title of Thesis/Project |
| firstname | Varchar | First name of Student Supervised |
| lastname | Varchar | Surname of Student Supervised |
| level | Int | Degree Level of Student |
| year | Varchar | Academic Year |

Table 13: Research Interests and Associations

|  |  |  |
| --- | --- | --- |
| Column Name | Data Type | Description |
| research | Varchar | Name of research field |
| interest | Varchar | interest |

Table 14: Conferences Attended

|  |  |  |
| --- | --- | --- |
| Column Name | Data Type | Description |
| title | Varchar | Name of Conference |
| year | Varchar | Academic Year |

Table 15: Technical Presentations

|  |  |  |
| --- | --- | --- |
| Column Name | Data Type | Description |
| title | Varchar | Title of Presentation |
| year | Varchar | Academic Year |

Table 16: University Professional Service Roles

|  |  |  |
| --- | --- | --- |
| Column Name | Data Type | Description |
| role | Varchar | Title of the Role |
| association | Varchar | Association of the Role |
| date | date | Date |

Table 17: Professional Consultancies

|  |  |  |
| --- | --- | --- |
| Column Name | Data Type | Description |
| title | Varchar | Title of Consultancy |
| position | Varchar | Position |
| period | Varchar | Time period |

Table 18: Professional Development

|  |  |  |
| --- | --- | --- |
| Column Name | Data Type | Description |
| title | Varchar | Title |
| year\_start | Int | Year Commenced |
| year\_end | Int | Year Ended |

Table 19: Contributions to the Department, Faculty or University

|  |  |  |
| --- | --- | --- |
| Column Name | Data Type | Description |
| contribution | Varchar | Contribution |
| year | Varchar | Academic Year |

Table 20: Other Professional Activities

|  |  |  |
| --- | --- | --- |
| Column Name | Data Type | Description |
| activity | Varchar | Other Activity |

Table 21: Request to the Administrator

|  |  |  |
| --- | --- | --- |
| Column Name | Data Type | Description |
| information | Varchar | Requested information |
| message | Varchar | Reason for Request |

Table 22: Contacts

|  |  |  |
| --- | --- | --- |
| Column Name | Data Type | Description |
| information | Varchar | Information |
| email | Varchar | Email Address |
| name | Varchar | Name of Contact |

### Data Flow Diagrams

The following dataflow diagrams show how the modules interact with each other and how information is passed between them to achieve the required functions of the system. A rectangle outlined in green represents the user, a circle outlined in blue represents the module itself and the arrows show the flow of the data while the red outlined shapes represent the database model. Figures 7-10 show the DFDs for the 3 main modules of the system.



Figure 7: Authentication Module DFD Level 1



Figure 8: Authentication Module DFD Level 2



Figure 9: Admin Module DFD Level 2



Figure 10: Staff Module DFD Level 2

### User Interface Design

As the system is required to be cloud-based, the user interface will be in the form of website pages which allow the user to access the functions of the system. There will be two main home pages for each type of user, the staff home page and the admin home page.

The staff home page will contain tabs for viewing and editing profile information, viewing contacts, requesting information, changing password, and logging out of the system.

The admin home page will contain tabs for viewing employees, adding accounts, generating annual reports, updating university contacts list, viewing requests submitted by staff users, changing password, and logging out of the system.

Each tab will render a new webpage when clicked displaying the requested information from the database to the user, allowing them to edit or add information permitted by the functions of the system.

## Design Implementation

The Django framework allows a system to be implemented in separate reusable components called apps. Each app is housed in its own folder in the project directory and has several auto created ‘.py’ files as shown in Figure 11.

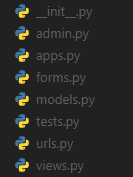


Figure 11: Default Django App Files

The ‘models.py’ file is used to create the database models to store required information. The ‘forms.py’ file allows forms to be created using the database models. These forms can then be rendered on html templates to all users to add and edit data from the database. The ‘admin.py’ file is used to register the created database models to the database. The ‘views.py’ file is where the main functions of the app are created. The return type of these view functions is usually an HTML template that the system will render to the user. These functions are then configured to be called by mapping them to specific URLs in the ‘urls.py’ file. The ‘tests.py’ file is used to write unit test functions for the created view functions. A separate app was created for each of the modules of the system architecture shown in Figure 6.

### HRIS\_APP

The ‘hris\_app’ was created as the authentication module of the system. The database object to store user information was created in the models.py file of this app. The Django framework contains a default user model called ‘AbstractUser’ however a custom model was created that inherits from this class to store the user type within this model. Separate models to contain all the staff users and the admin users separately were also created, using the CustomUser model with the ‘OnetoOneField’ attribute. This specified a one-to-one relationship between the CustomUser instance and the StaffUser or AdminUser instance. Thus, each admin and staff user will have an associated entry in the CustomUser database table. This is shown in Snippets 1 and 2 (See Appendix C).

Django also contains built in authentication functions to handle logging in and logging out of users. These were imported from the ‘django.contrib.auth’ library.

Register Function

The register function was defined in the views.py file of the hris\_app. Django view functions by default accept an HTTP request object as their input. This object contains the information of the user that sent the request as well as any other form information submitted from the HTML document that sent the request. The request object sent to this function contains the submitted first name, surname, email address and password of the user. The function performs validation on the submitted information to determine if the email address is already in use and returns a notification to the user in this case. Once the validation checks are passed the system creates a CustomUser Employee and StaffUser object with the information submitted by the user and redirects the user to the login page. This function can be seen in Snippet 3 (Appendix C).

Login Function

The log in function accepts a request containing the email address and password submitted by the user and firsts checks if the email address entered exists in the custom users model in the database. If the user exists, the function verifies the password using Django’s authentication function and then the login function logs the user into the system and redirects them to the staff page or the admin page depending on the user type. This implemented function can be seen in Snippet 4 (Appendix C).

Add Account Function

This function allows admin users to add a new staff or admin account. The admin user will input the email address, first name, last name and select the account type. The function then verifies that the email address is valid and not already associated with an existing account. The CustomUser instance will then be created with the default password of ‘password’ and if the account specified was a staff account, an instance in the employee information database model will be created and a success message will be displayed. The implementation of this function is shown in Snippet 5 (See Appendix C).

Batch Add Staff Accounts Function

This function allows an admin user to upload a ‘.txt’ file with email addresses of employees for staff accounts to be created. This function decodes the file and reads the emails in the file, verifies that each is valid and is not currently associated with an existing staff account. Custom User and Employee objects will be created for each of the valid email addresses with the default password of ‘password.’ The email addresses that are not valid will be displayed to the user so than these can be altered and entered again for accounts to be created. This function is shown in Snippet 6 (See Appendix C).

Reset Staff Account Function

This function allows the admin user to reset a staff account. This function deletes the specified CustomUser model for the selected employee. This in turn deletes all other database model entries that had the id of this CustomUser instance as a foreign key. A foreign key is a database attribute that established a many to one link between tables in relational databases. A new CustomUser instance is then created with the default password of ‘password’ and a new Employee database model object. A success message is then displayed, and the user is redirected to the admin view homepage. Snippet 7 shows the implementation of this function.

The implementation of the Reset Staff Account Function along with the Add Account and Batch Add Staff Accounts function serve to complete objective 4 (a). Other functions of the hris\_app include the ‘home’ function which displays the homepage of the system where users can either login or register. The ‘logout\_user’ function is used to log the user out of the system. Django contains built in functions for changing and resetting passwords of users from the ‘auth\_views’ module so these functions were implemented to complete objective 4 (b). The code for these functions can be found in the views.py file in the hris\_app folder of Appendix D (3020 Appendices Upload).

### STAFF\_APP

The ‘staff\_app’ app was created to serve as the staff module of the system. Within it, the database models of Tables 5-22 were created. Snippet 8 shows the creation of the Awards model. The ‘user’ field uses the ‘ForeignKey’ attribute associating the model with the CustomUser model. Thus, a user can have multiple awards associated with them. The other created models were defined with this same user field. The code implementing the other created models can be found in the models.py file in the staff\_app folder of Appendix D (3020 Appendices Upload).

The created database models and how they relate to each other is shown in Figure 12.



Figure 12: ER Diagram for Database Models

In order for users to add, edit and delete entries in these models, Django model forms were created for each. These forms can be passed as context to an html template where they are rendered to the user. Snippet 9 (See Appendix C) shows the code for the form used to create a publication entry.

All forms can be found in the forms.py file in the staff\_app folder of Appendix D (3020 Appendices Upload). To utilise these forms, view functions to create a new entry, edit an existing entry, delete an existing entry, and view existing entries associated with the current user were created.

View Entries

The functions to view the entries associated with the current user uses the Django ‘.objects.filter()’ method to obtain all the model objects related to the current user. This filter method returns a query set data type. This query set is then passed to an HTML template to be rendered to the user. The implementation for the function to view all publications of the current user is shown in Snippet 10 (See Appendix C).

Add Entry

The functions to add an entry to a database model initialises an instance of the respective model creation form and renders it to the user. When the user submits the form, the information is validated and an entry in the model is created, saving the entered information with the users id so that a relationship can be created within the model linking it to the CustomUser model. The function then displays a success message to the user and the user is redirected to the view entries page. The implementation for the function to add a publication is shown in Snippet 11 (See Appendix C).

Edit Entry

The functions to edit an existing model entry retrieves the object from the model by the id passed to the function and renders this entry to the associated model form for the user to edit. When the user submits the form, the information is verified before a success message is displayed and the user is redirected to the associated view entries page. Snippet 12 (See Appendix C) shows the implementation of the function to edit a publication.

Delete Entry

The functions to delete an entry form a specific model by the id passed to the function. It then deletes the entry and displays a success message to the user. The user is then redirected to the view entries page. Snippet 13 (See Appendix C) shows the implantation of the function to delete an entry from the publications model.

These implemented functions of the staff\_app module served to achieve objective 4 (d) allowing staff users to directly enter and update essential data to the system. The definition of all other view, add edit and delete functions for each respective database model can be found in the views.py file in the staff\_app folder of Appendix D (3020 Appendices Upload).

Requests for Information Function

This function allows a staff user to submit a form to request information from the admin users. The fields of information to be requested by the staff user include information on Study and Travel Grants, Institutional Visit Allowances, Development and Training Grants, Book Grants, Pension Plans, Claim Forms and it also includes an ‘other’ field. The staff user may also include an optional message with the request. Upon submission, a success message is displayed to the user. The code implementation of this function can be found in the views.py file in the staff\_app folder of Appendix D (3020 Appendices Upload). The implementation of this function serves to achieve objective 4 (f).

### ADMIN\_APP

This app serves as the admin module of the system. Here admin users will be able to view employee entered information and generate reports based on this information. The admin users will also have access to the add and reset accounts functions of the hris\_app.

View All Employees Function

This function uses the Django ‘.objects.all()’ method on the employee database model to obtain all the objects from this table. It then uses Django’s built in ‘Paginator’ function to slip this query set into sections containing 10 objects each for displaying purposes. This function also contains a search function allowing users to search the list of employees. This search function uses a regular expression to replace any spaces in the search query and replaces it with a period. It then compares this string with the email addresses of the employees and returns a query set of filtered objects whose email addresses contain part of the search query. The implementation of this function is shown in Snippet 14 (See Appendix C). Similar functions to view all publications were also created and these can be found in the views.py file of admin\_app folder of Appendix D (3020 Appendices Upload).

View Employee Function

This function accepts a user id and calls a custom function called ‘get\_user\_info().’ This custom function returns a dictionary of query sets containing all associated information from all database models. This dictionary is the passed to an HTML template by the view employee function to be viewed by the admin user. The view all employees function and the get\_user\_info() function can be seen in Snippets 15 and 16 respectively (See Appendix C).

Annual Reports Function

This function accepts 3 search inputs; first name, last name and academic year. The functions checks if a user exists with the given first name and last name. If a user does not exist, an error message is displayed. If the user exists, the function obtains all publications, presentations, awards, honours, and contributions to the department associated with the searched user. This information is typically required for the evaluation of staff members. This information is passed via a dictionary to an HTML template to display this information to the user. This information can then be downloaded in PDF format. This function along with the employee information function serves to achieve objective 4 (e). The implementation of this function can be seen in Snippet 17 (See Appendix C).

Annual Reports PDF Function

This function utilises the python ‘report-lab’ library to dynamically create a pdf file containing the information of an employee obtained from the annual reports function. This pdf file can then be downloaded to the admin user’s computer. The code implementation of this function can be found in the views.py file in the admin\_app folder of Appendix D (3020 Appendices Upload). This function along with annual reports function serves to achieve objective 4 (e).

Update Contacts Function

This function allows an admin user to update entries in the contacts model. The contact information input by the admin user is displayed to the staff user at their request. The code implementation of this function can be found in the views.py file in the admin\_app folder of Appendix D (3020 Appendices Upload).

### User Interface

The user interface was implemented in the form of multiple HTML templates utilising HTML, CSS and python language. The templates are called by the view functions and rendered out to the user. Since the design of the UI was not a specific requirement of the system, the creation of these templates will not be explained in detail but they can all be found in the ‘templates’ folder of Appendix D. (3020 Appendices Upload).

### Other Functions

Automatic Requests for Information Function

The Django APScheduler library was used to assist this function. APScheduler runs in the background of the system and allows tasks to be automatically scheduled to run. The implemented function gets a list of all user ids. It then iterates through the list of ids and checks every database model for an entry with the id. If in any database model no entries exist, the function automatically sends an email to the email address associated with the id detailing the information missing. The email is sent utilising the Django ‘send\_mail()’ function. This function sends an email using an SMTP backend configured in the Django application’s ‘settings.py’ file. The configuration for this is shown in Snippet 18 (See Appendix C). APScheduler was used to assign this function once a week. An example of a part of this function can be found in Snippet 19, while the entire function can be viewed in the jobs.py file in the jobs folder of Appendix D (3020 Appendices Upload). The Automatic Requests for Information Function was implemented to complete objective 4(c).

Auto Logout

The ‘django\_auto\_logout’ library was implemented to automatically log out a user and return them to the home page after 10 minutes of inactivity. This was configured by creating a ‘AUTO\_LOGOUT’ variable in the ‘settings.py’ file of the system. This variable and its configurations can be seen in Snippet 20 (See Appendix C).

### Hosting

One of the core requirements of the system is that the implemented software be cloud-based as opposed to hosted on a dedicated machine. Thus ‘PythonAnywhere’ was used to host the completed system so that it can be accessed by all the required members of staff form their computers. PythonAnywhere is cloud-based hosting service that specialises in python-based applications. It offers a free tier that offers 512mb of database storage. This is sufficient for the scale of this particular application however if the application grows in the future this free tier can easily upgraded to meet the scaling needs of the system.

# Results



## User Interface

Graphical user interface

Description automatically generated

Figure 13: Login Page

Graphical user interface

Description automatically generated

Figure 14: Registration Page

### Staff Pages

Graphical user interface, website

Description automatically generated

Figure 15: Staff Home Page

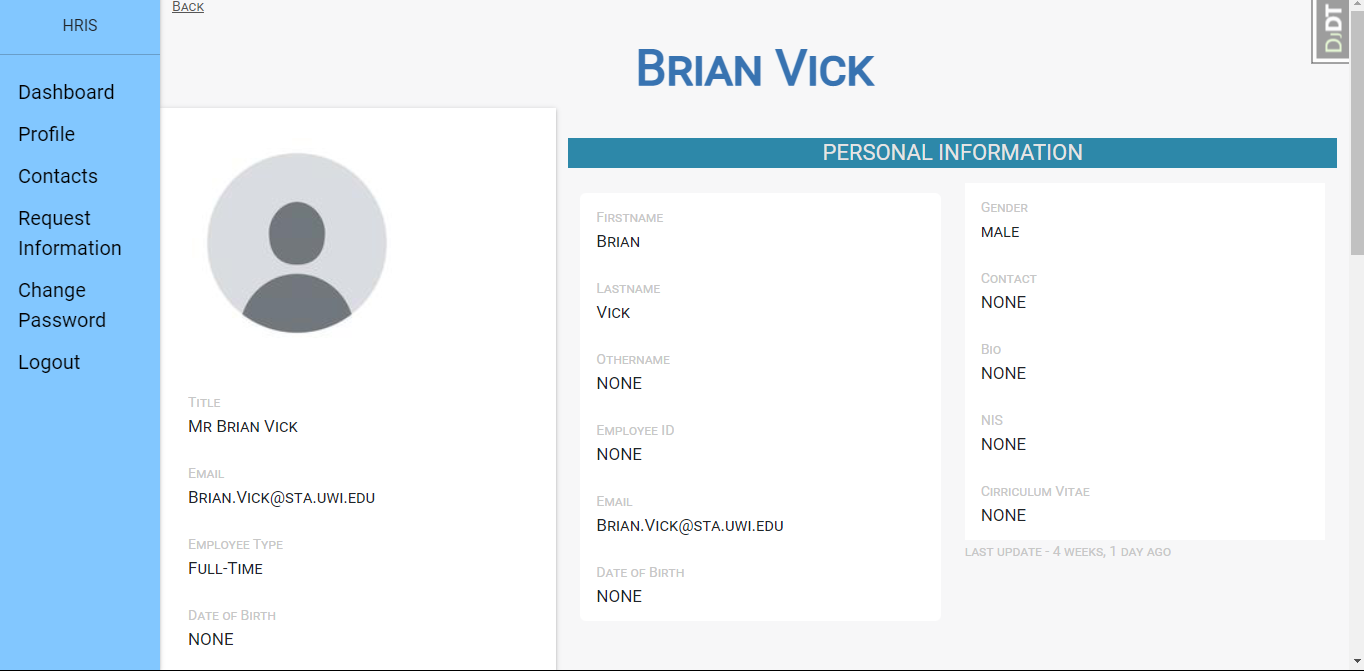


Figure 16: Staff Profile Page

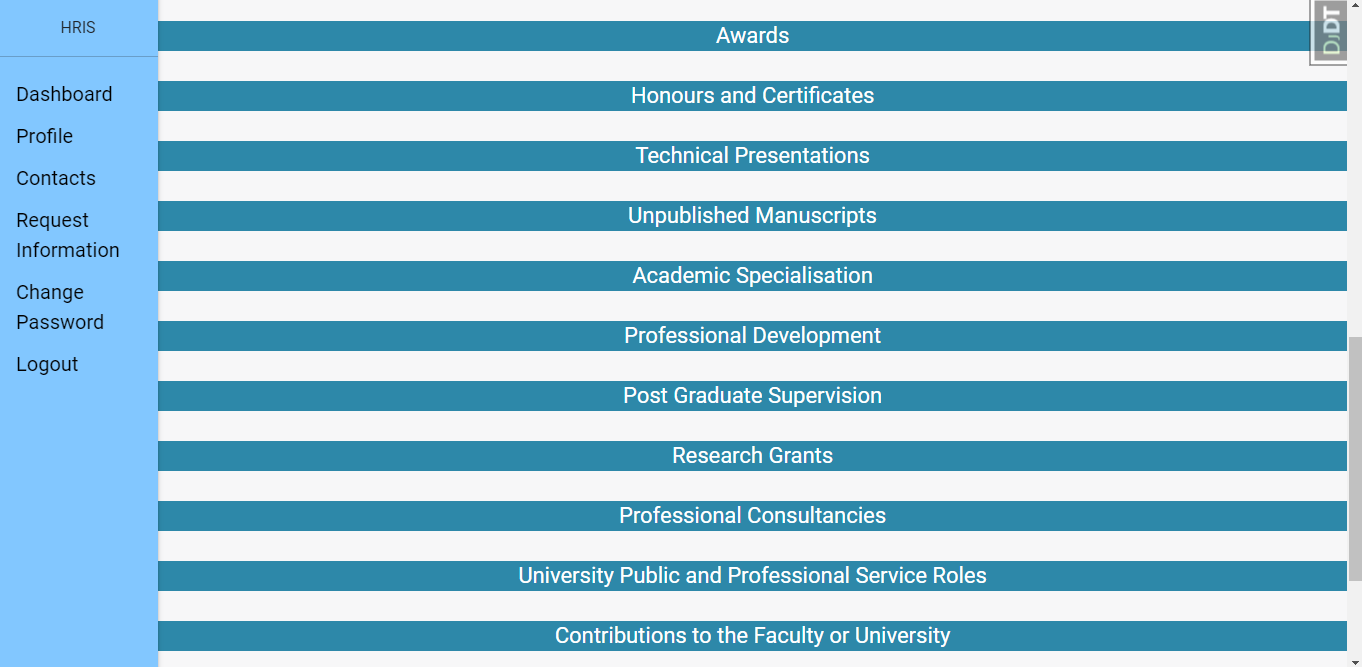


Figure 17: Staff Profile Page Continued

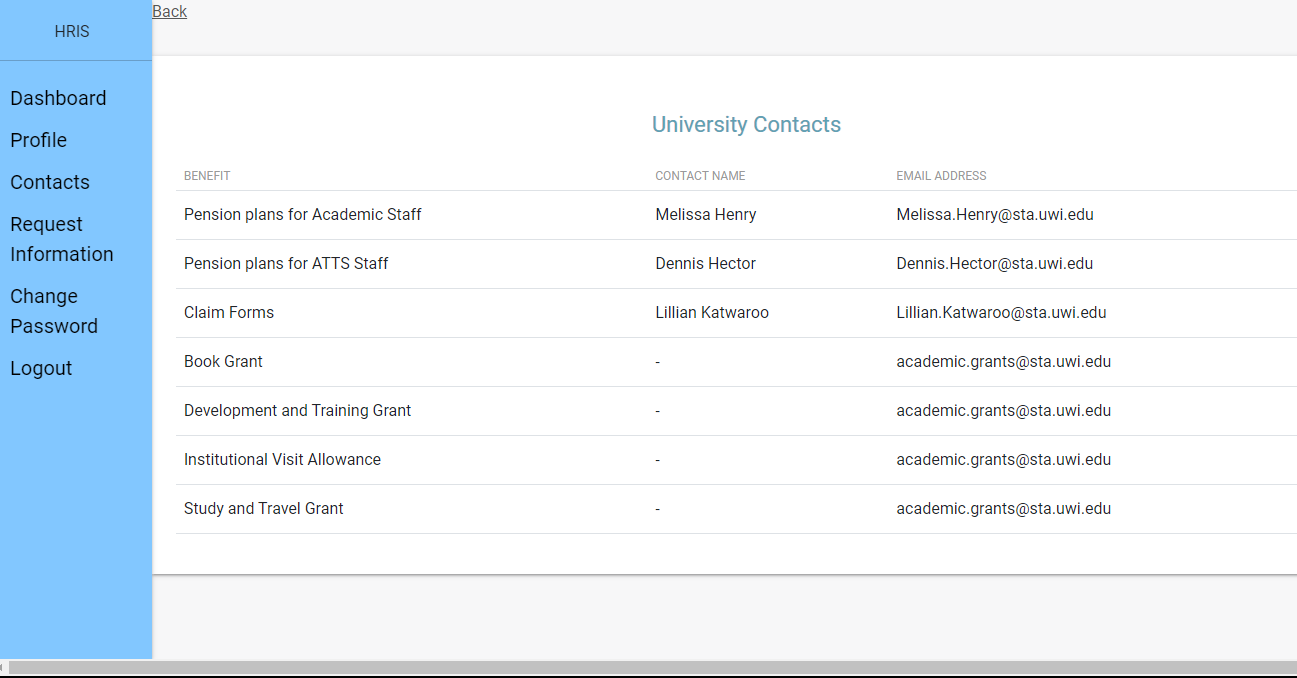


Figure 18: Staff View Contacts Page

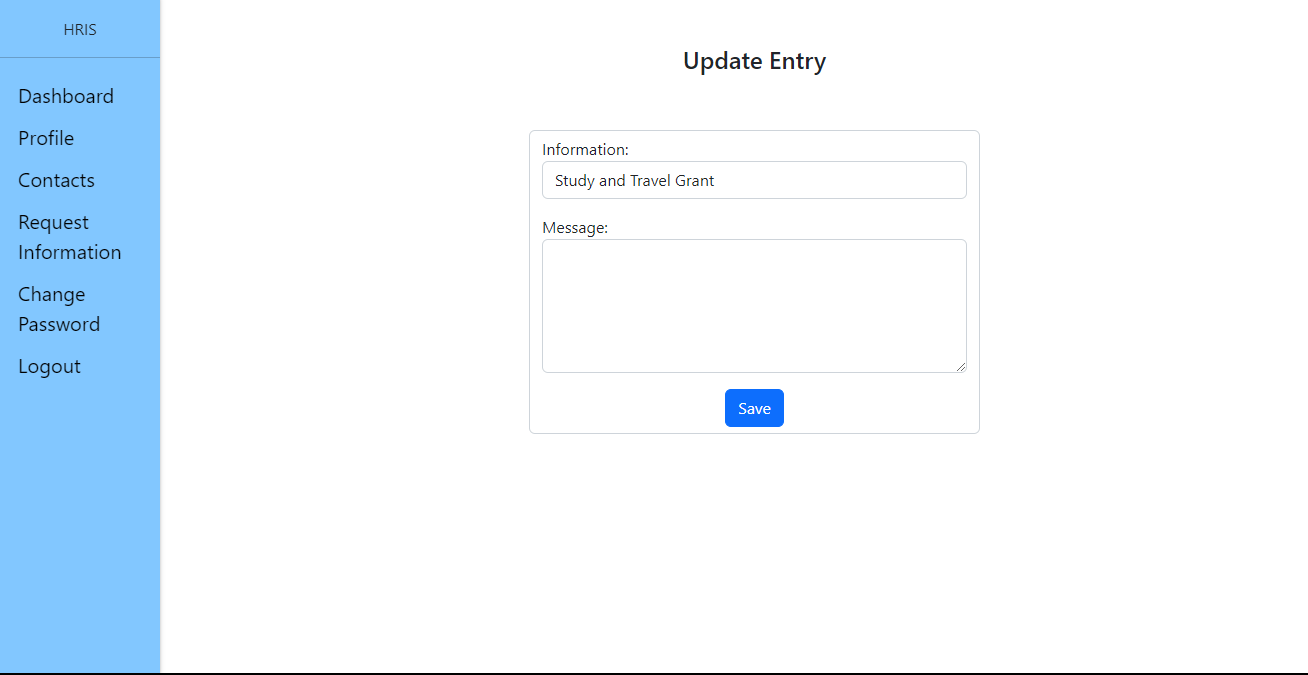


Figure 19: Staff Submit Request For Information Page

### Admin Pages

Graphical user interface, application, website

Description automatically generated

Figure 20: Admin Home Page

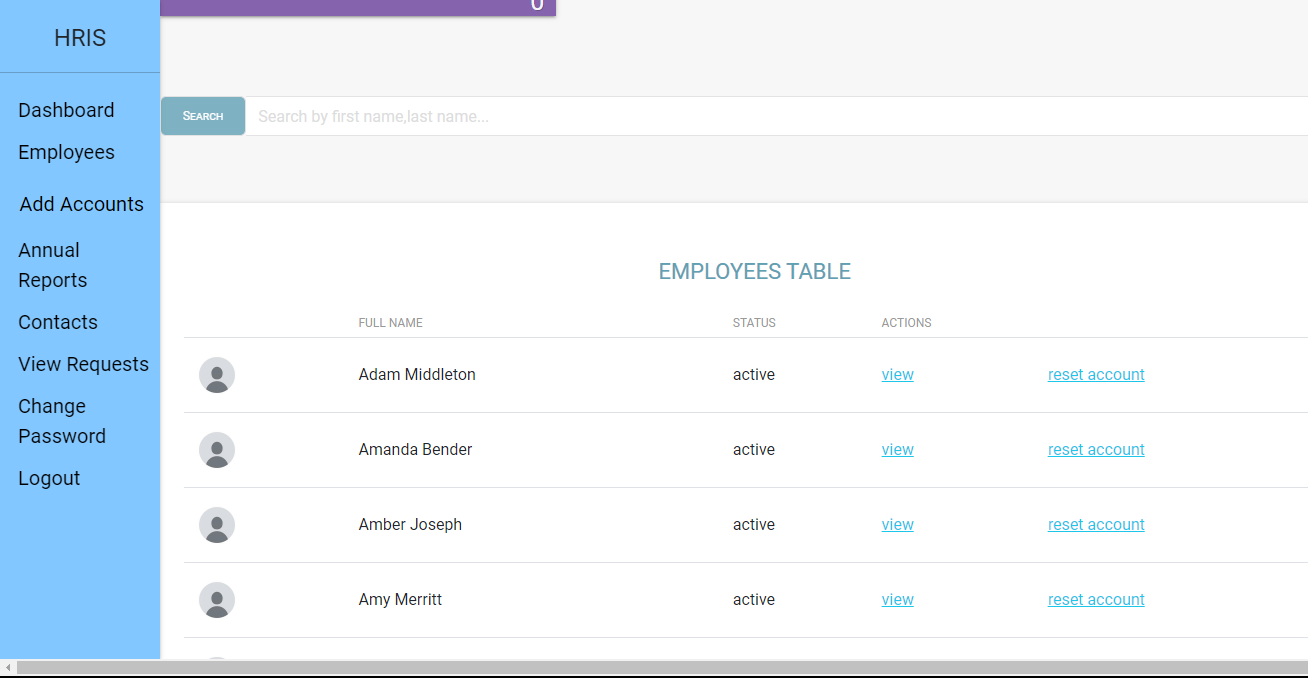


Figure 21: Admin View Employees Page

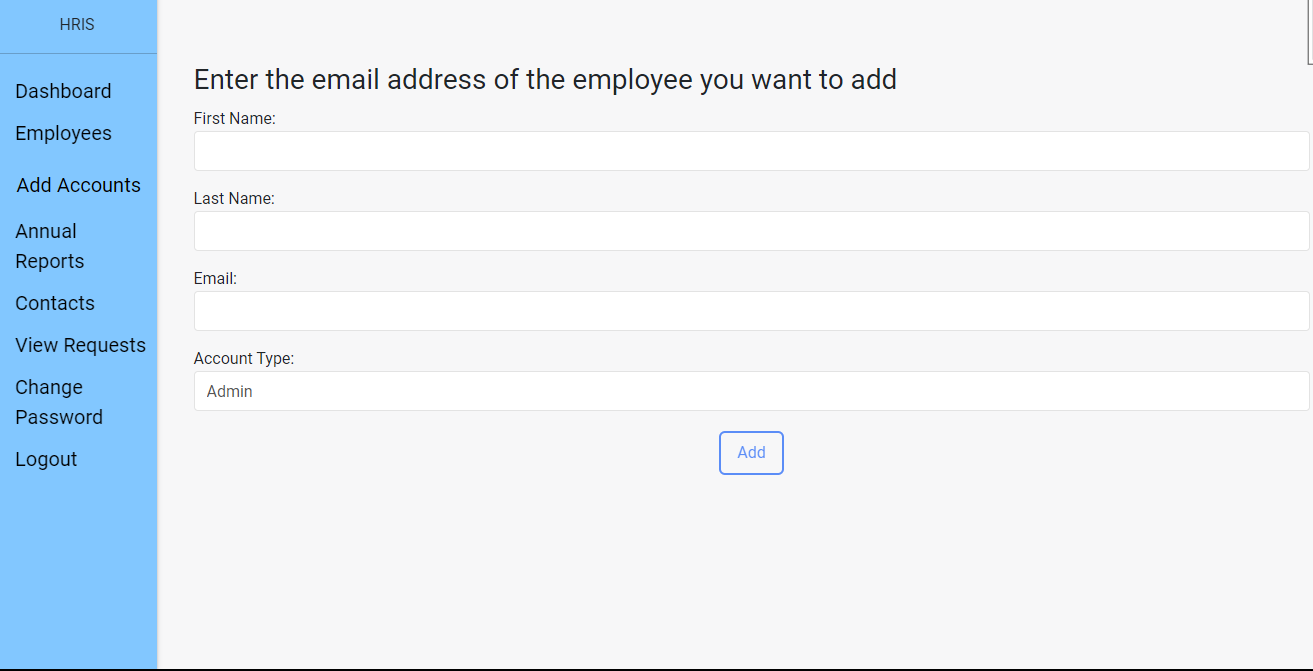


Figure 22: Admin Add Account Page

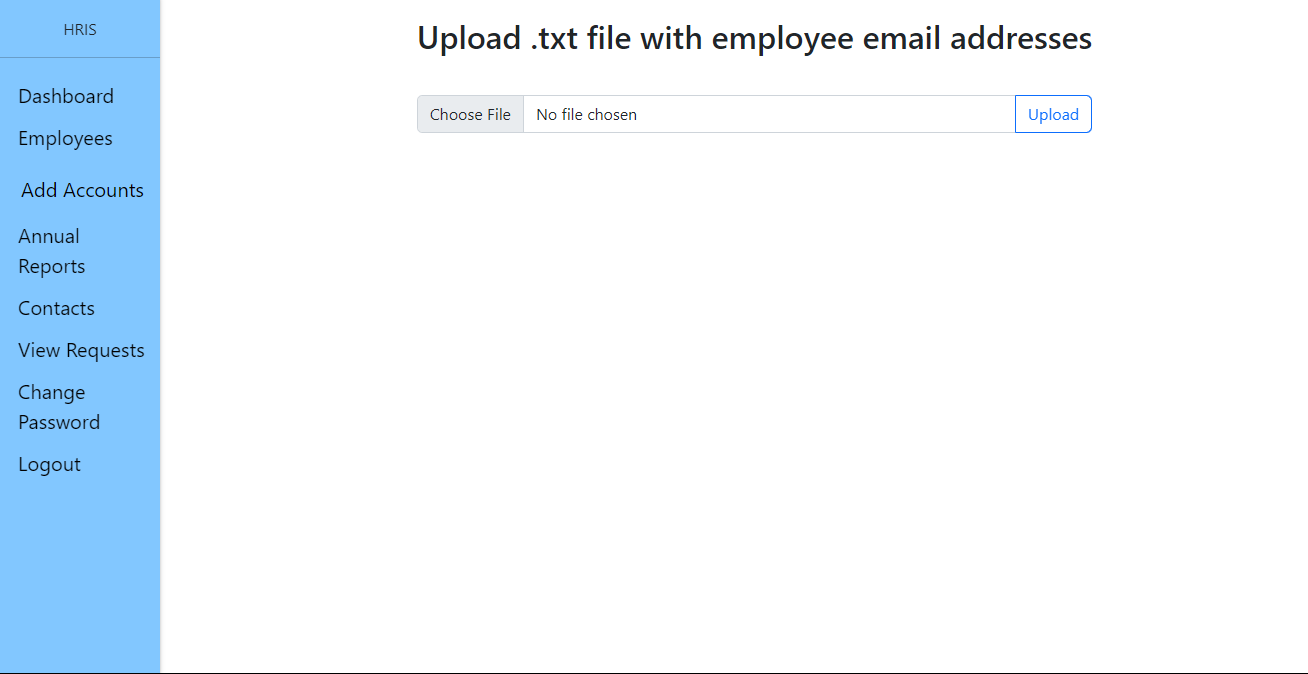


Figure 23: Admin Batch Add Accounts Page

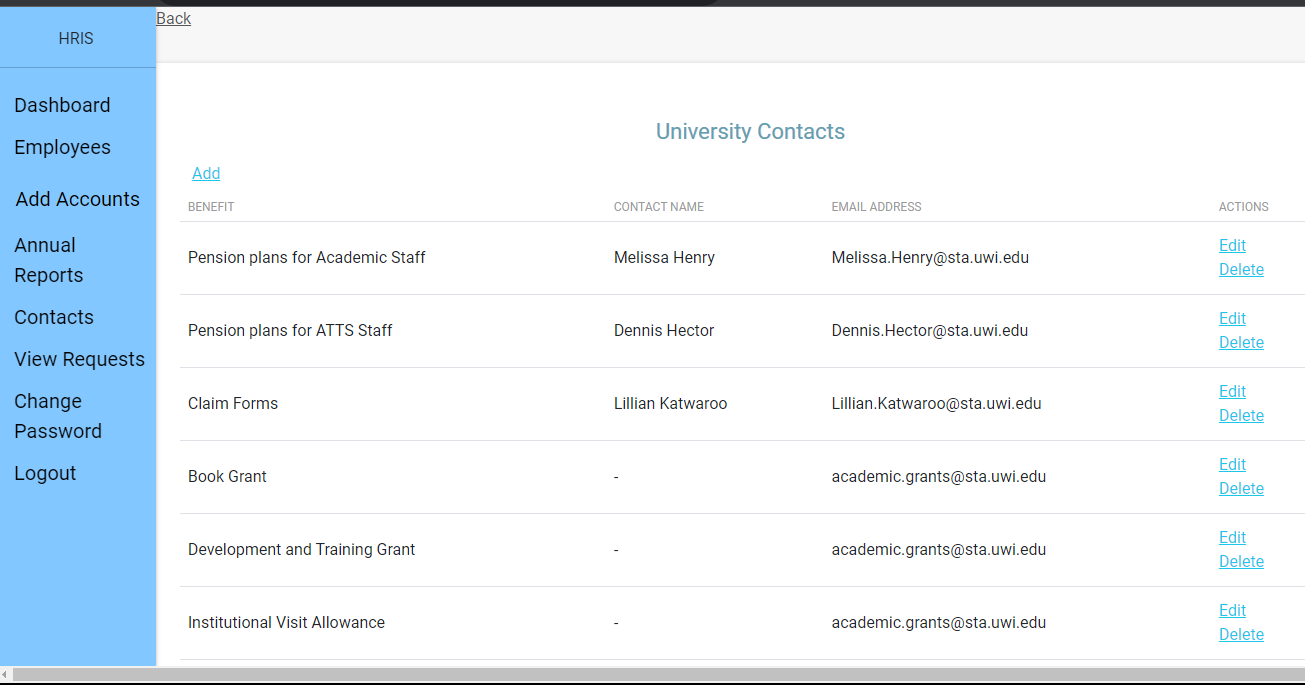


Figure 24: Admin View Contacts Page

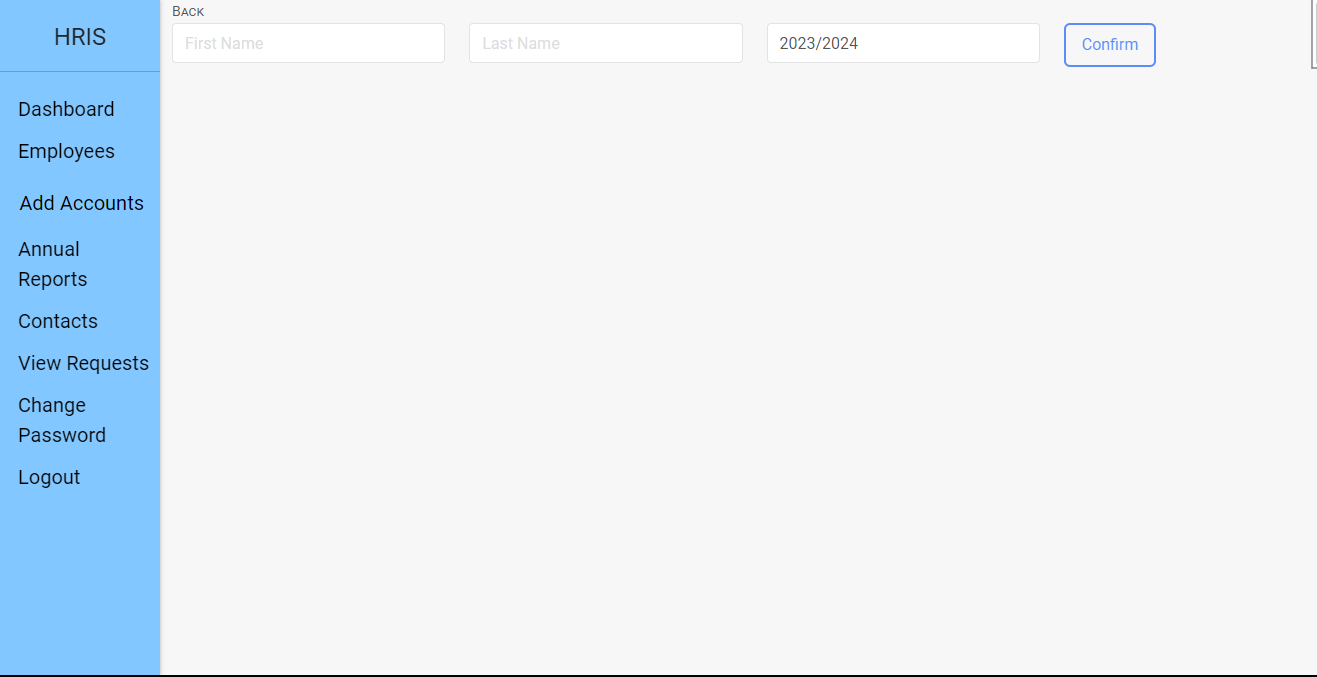


Figure 25: Admin Annual Reports Page 1/3

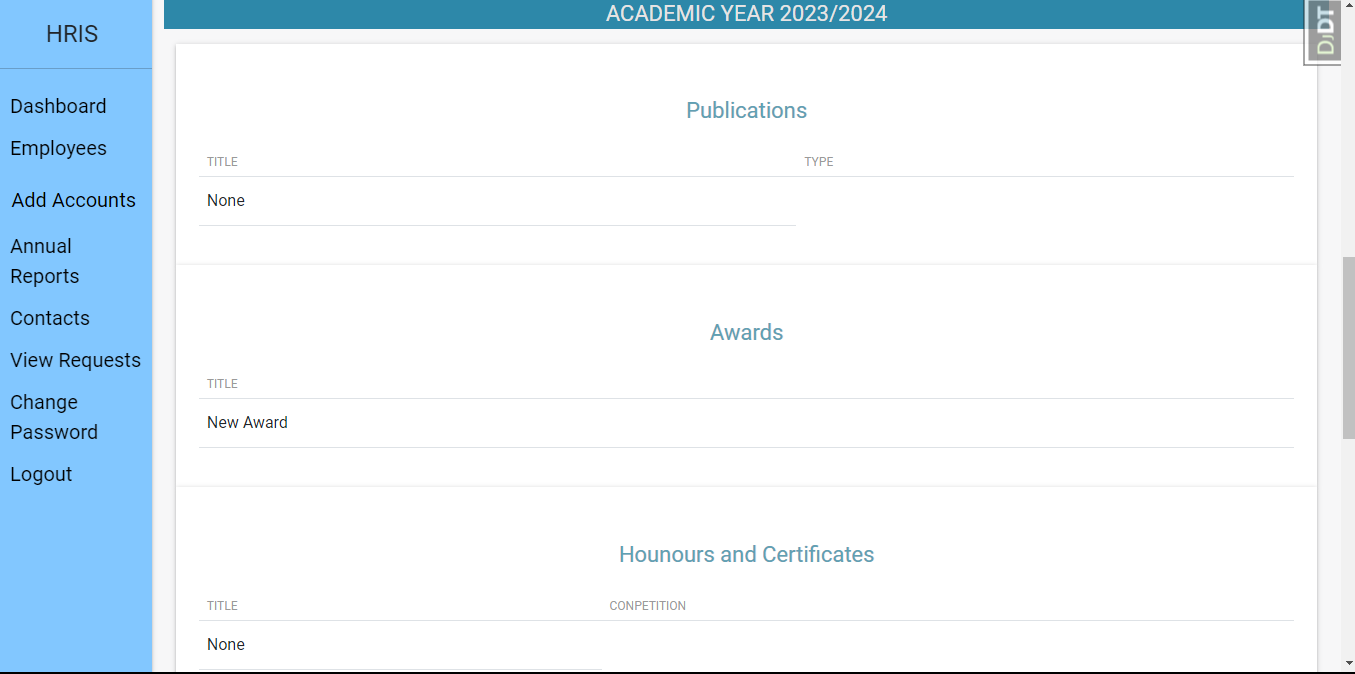


Figure 26: Admin Annual Reports Page 2/3

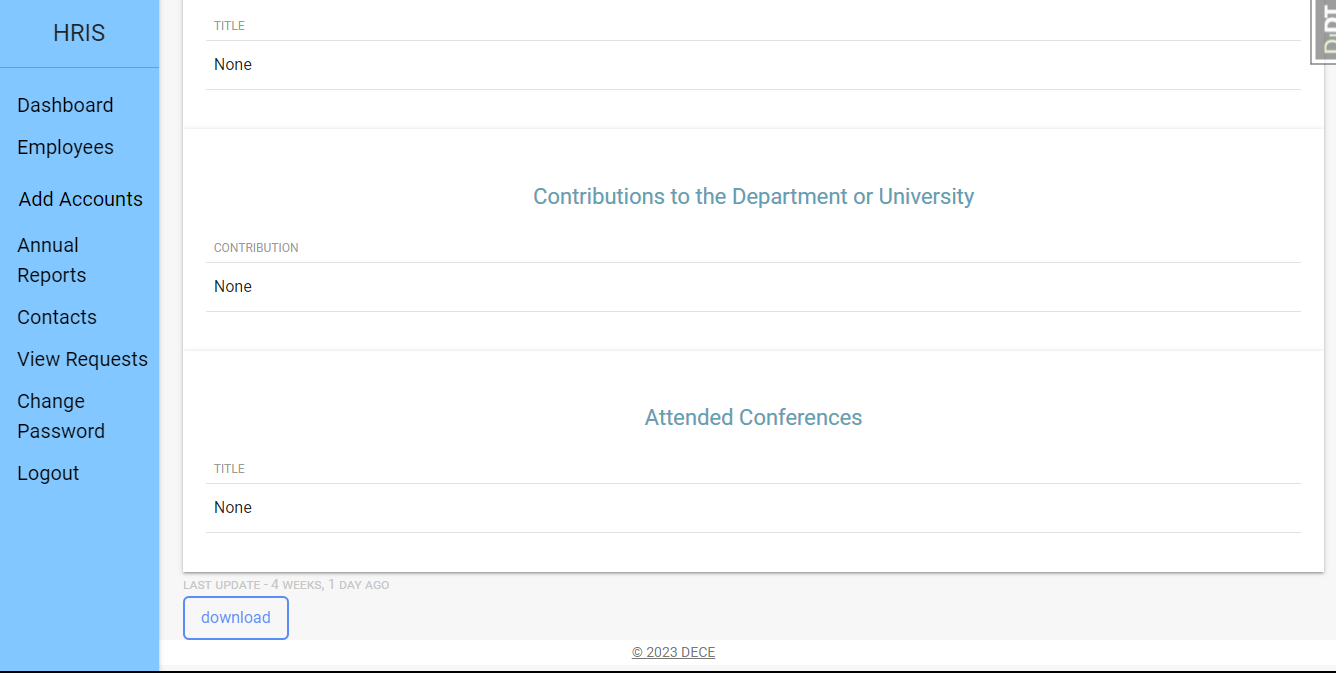


Figure 27: Admin Annual Reports Page 3/3

## Test Results

The designs and functions outlined in Section 3.5 were implemented using the Django Framework. Testing was subsequently performed using the ‘TestCase’ library native to the framework. This library allows the simulation of a user interacting with the system and accessing the various functions. The mapping of the URLs to the Django view functions was tested first to ensure that when a user requests a webpage the system will redirect the user to the correct URL and perform the correct function. All tests were passed. Documentation of this can be found in Appendix B. Unit testing was then performed on the view functions of the system for each app to analyse how the system behaves with more specific test cases.

### HRIS\_APP

Tables 23-26 show the test results for the functions of the hris\_app. The full test case documentation can be found in Appendix B.

Table 23: Registration Function Test Results

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Test Case | Steps | Expected Results | Status |
| 1 | Successful Registration | Input first name, last name, valid email address, password, matching confirmation password, | CustomUser object created.  Employee object created.  StaffUser object created.  User is redirected to login page. | Pass |
| 2 | Missing Information | Leave one or more fields blank, | Error message displayed to the user. | Pass |
| 3 | Password Mismatch | Input first name, last name, valid email address, password, non- matching confirmation password, | Error message displayed to the user. | Pass |

The Registration Function’s first test case simulated a register request with all required information to ensure that the function created a CustomUser model, an Employee model and a StaffUser model successfully. The second test case simulated a register request with required information fields empty to ensure that the function will return error. The third test case simulated a registration request with the password and confirm password fields not matching to ensure that the function will return the appropriate error request to the user.

Table 24: Login Function Test Results

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Test Case | Steps | Expected Results | Status |
| 1 | Valid Login | Input valid login credentials | User is redirected to either admin or staff homepage depending on the user type. | Pass |
| 2 | Invalid Login | Input invalid login credentials | Error message is displayed to the user. | Pass |

Test case 1 simulated a login request where the username and password field matched a stored CustomUser object in the database and the function correctly redirects the user to the staff home page or the admin home page depending on the type of the user. The second test case simulated a login request with invalid login credentials. A login error was displayed to the user and the user was successfully redirected to the login page.

Table 25: Add Account Function Test Results

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Test Case | Steps | Expected Results | Status |
| 1 | Add Account Authenticated | Attempt to access function while logged in with an admin account. | Add Account Page rendered to user. | Pass |
| 2 | Add Account Unauthenticated | Access function while not logged in | User redirected to login page. | Pass |
| 3 | Duplicate Email | Access function while logged in with an admin account.  Enter email address that is already in use | Error displayed and user is redirected to the Add Account page. | Pass |
| 4 | Invalid Email | Access function while logged in with an admin account.  Enter email address that does not end with “sta.uwi.edu” | Error displayed and user is redirected to the Add Account page. | Pass |

Test case 1 simulated an authenticated admin user attempting to access the add account feature. The user was successfully redirected to the Add Accounts Page. Test case 2 simulated an unauthenticated user attempting to access the function and the user was successfully redirected to the login page. Test case 3 simulated an authenticated admin user entering an email address that is already associated with an existing user. The system successfully returned an error to the user. Test case 4 simulated the input of an email address to the function that does not end with ‘sta.uwi.edu.’ An error was successfully displayed to the user.

Table 26: Batch Add Function Test Results

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Test Case | Steps | Expected Results | Status |
| 1 | Txt file Input | Access function while logged in with an admin account.  Upload .txt file containing valid emails | CustomUser, StaffUser and Employee objects are created for each email in file and a success message is displayed to the user. | Pass |
| 2 | Non Txt file Input | Access function while logged in with an admin account.  Upload a non .txt file | Error displayed and user is redirected to the Batch Add Account page. | Pass |
| 3 | Txt file contains Invalid emails | Access function while logged in with an admin account.  Upload .txt file containing invalid emails | Error displayed showing all invalid email addresses and user is redirected to the Batch Add Account page. | Pass |

Test case one simulates a user uploading a .txt file to the batch add function. The function accepted the file and created accounts for all the valid email addresses contained in the file. Test case 2 simulated a non txt file being input to the function. The function returned an error to the user. Test case 3 simulated a txt file with invalid email addresses being input to the function. The function created the CustomUser, Employee and StaffUser objects for the valid email addresses while it returned a list of all invalid email addresses.

### STAFF\_APP

Tables 27-30 show the test results for the functions of the staff\_app. The full test case documentation can be found in Appendix B. All of the view, edit, add and delete functions of the staff\_app work in the same way and thus they were all tested in a similar way. The results shown below will stand for theses grouped functions as a whole.

Table 27: View Model Data Functions Test Results

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Test Case | Steps | Expected results | Status |
| 1 | Authenticated user | Access function while logged in | View page is rendered to the user. | Pass |
| 2 | Unauthenticated user | Access function while not logged in | User is redirected to the login page. | Pass |

Test case 1 simulates using the function when the user is authenticated. The function correctly displays the information to the user. Test case 2 simulates an unauthenticated user attempting to view the information displayed by the function. The function correctly redirects the user to the login page.

Table 28: Add Model Object Functions Test Results

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Test Case | Steps | Expected Results | Status |
| 1 | Valid information | Access function while logged in.  Submit valid information into the form fields. | New object added to model with submitted information.  Success message displayed to user. | Pass |
| 2 | Invalid Information | Access function while logged in.  Submit invalid information into the form fields. | Error displayed to user.  User is redirected to Add object page. | Pass |

Test case 1 involved adding valid information into the fields of the respective create form for the model. The function correctly saved the data to the model in the database. Test Case 2 involved adding information into the form field that was invalid. The function returned an error, and no new object was added to the respective database model.

Table 29: Edit Model Object Functions Test Results

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Test Case | Steps | Expected Results | Status |
| 1 | Valid object id | Access function while logged in.  Submit request with valid object id to be edited | Functions retrieves object with id equal to the requested id and user is allowed to edit object. | Pass |
| 2 | Invalid object id | Access function while logged in.  Submit request with invalid object id to be edited | 404 error displayed to user. | Pass |

Test case 1 involved passing a valid object id to be retrieved and edited. The function retrieved the correct object corresponding to the given id and saved the new information when the form was submitted. Test case 2 involved passing an id that does not exist within the model. The function correctly returns and error to the user notifying them that the object does not exist.

Table 30: Delete Model Object Functions Test Results

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Test Case | Steps | Expected Results | Status |
| 1 | Valid object id | Access function while logged in.  Submit request with valid object id to be deleted | Functions retrieves object with id equal to the requested id and deletes object from database. | Pass |
| 2 | Invalid object id | Access function while logged in.  Submit request with invalid object id to be deleted | 404 error displayed to user. | Pass |

Test case 1 involved passing a valid object id to be retrieved and deleted. The function retrieved the correct object corresponding to the given id and deleted the object from the respective model. Test case 2 involved passing an id that does not belong to an object within the model. The function correctly returns and error to the user notifying them that the object does not exist.

### ADMIN\_APP

Tables 31-33 show the test results for the functions of the hris\_app. The full test case documentation can be found in Appendix B.

Table 31: View All Employees Function Test Results

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Test Case | Steps | Expected Results | Status |
| 1 | Authenticated admin user | Access function while logged in to admin account | View Employees page is displayed to the user | Pass |
| 2 | Unauthenticated user | Access function while not logged in | User is redirected to the login page. | Pass |

Test Case 1 displayed the employees list to the user while test case two redirected the user to the login page.

Table 32: View Employee Function Test Results

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Test Case | Steps | Expected Results | Status |
| 1 | Authenticated admin user | Access function while logged in to admin account | View Employees page is displayed to the user | Pass |
| 2 | Unauthenticated user | Access function while not logged in | User is redirected to the login page. | Pass |

Test case 1 successfully displayed the profile of the selected employee to the user.

Test case 2 redirected the user to the login page.

Table 33: Annual Reports Function Test Results

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Test Case | Steps | Expected Results | Status |
| 1 | Valid first name and last name | Access function while logged in to admin account.  Submit valid employee first name and last name to function | Report with employee information for given academic year is rendered to the user. | Pass |
| 2 | Invalid first or last name | Access function while logged in to admin account.  Submit invalid employee first name and last name to function | Error is displayed to the user. The user is redirected to the Annual Reports page. | Pass |

Test case 1 involved passing the valid first and last name of an existing employee. The function returned the associated employee information with the information associated with the selected academic year. Test case 2 involved passing an invalid first and last name to the function. The function returned an error to the user and no report was generated.

# Conclusion

The HRIS system was successfully designed and implemented with an appropriate software package allowing the system to be robust and scalable. Each system module underwent rigorous testing and passed all cases. The final results indicate a system that achieved all functional objectives outlined in the SRS (See Appendix A) and also contained additional features such the automatic timeout functionality, the ability to download the created reports in PDF format and extra security features inherited from the Django framework. Objective 1 was achieved in Section 2. Objectives 2 and 3 were achieved in Appendices A and B, objective 4 was achieved in Section 3, and objectives 5, 6 and 7 were achieved Section 4. The system also has the ability to be upgraded in the future to accommodate more users by upgrading the hosting plan with the PythonAnywhere host. Another upgrade to the system would be the changing of the database form SQLite to either a MySQL or PostgreSQL as both these databses are better suited when a larger database is required with a very high number of simultaneous users.

# References

Barisic, A. F., M. Tomic, and M. Pejic Bach. 2022. “Adoption of Human Resource Information Systems: Impact of Industry, Size and Market.” *45th Jubilee International Convention on Information, Communication and Electronic Technology.* Opatija: IEEE. 1197-1202. doi:10.23919/MIPRO55190.2022.9803480.

Boehm, Barry W. 1988. “A Spiral Model of Software Development and Eenhancement.” *Computer.* IEEE. 61-72. doi:10.1109/2.59.

Chauhan, Akansha, Sanjeev Kr Sharma, and Tarun Tyagi. 2011. “Role of HRIS in improving modern HR operations.” *Review of Management, vol 1* 1 (2): 58-70.

Dyachenko, Anastasia. 2022. *Best Backend Technologies & Languages For Your Project.* 11 July. https://cadabra.studio/blog/best-backend-technologies-list-comparison-examples/.

Foundation, Django Software. 2022. “Django Documentation.” *Django.* 3 August. Accessed January 9, 2023. https://docs.djangoproject.com/en/4.1/.

Foundation, Python Software. 2023. *Python 3.10.10 documentation.* 10 February. Accessed February 10, 2023. https://docs.python.org/3.10/index.html.

GeeksforGeeks. 2022. “Unified Modeling Language (UML) | An Introduction.” *GeeksforGeeks.* 15 December. Accessed January 12, 2023. https://www.geeksforgeeks.org/unified-modeling-language-uml-introduction/.

H.A.S.S, Hewage, Hettiarachchi K.U, Jayarathna K.M.J.B, Hasintha K.P.C, Janaka L. Wijekoon, and Amila Senarathne. 2020. “Smart Human Resource Management System to Maximize Productivity.” *International Computer Symposium (ICS).* IEEE. 479-484. doi:10.1109/ICS51289.2020.00100.

Jacobson, Ivar, Grady Booch, and James Rumbaugh. 1999. *The Unified Software Development Process.* Addison Wesley Professional.

Korsun, Julia. 2022. *Why We Use Django Framework & What Is Django Used For.* 30 December. https://djangostars.com/blog/why-we-use-django-framework/.

Kumar, M. Rudra, Avinash Sharma, Y. Krishna Bhargavi, and G. Ramesh. 2022. “Human Resource Management Using Machine Learning-Based Solutions.” *3rd International Conference on Electronics and Sustainable Communication Systems.* IEEE. 801-806. doi:10.1109/ICESC54411.2022.9885526.

Liu, Kun, Jinmin Jiang, Xiaohan Ding, and Hui Sun. 2017. “Design and Development of Management Information System for Research Project Process Based on Front-End and Back-End Separation.” *2017 International Conference on Computing Intelligence and Information System (CIIS).* Nanjing: IEEE. 338-342. doi:10.1109/CIIS.2017.55.

Liu, Li. 2021. “Design of Human Resource Management Information System Based on Decision Tree Algorithm.” *2021 Global Reliability and Prognostics and Health Manageme.* IEEE. 1-6. doi:10.1109/PHM-Nanjing52125.2021.9613090.

Ma, Wenjing. 2021. “Construction of Intelligent Human Resource Management Information System Based on Python and Tree Node Mining.” *5th International Conference on Electronics, Communication and Aerospace Technology (ICECA).* IEEE. 1571-1574. doi:10.1109/ICECA52323.2021.9675873.

Mou, Nian, and Hao Luo. 2021. “Research and Computer Aided Design of Information System for Public Sector Using Database System and Digitization.” *IEEE International Conference on Data Science and Computer Application (ICDSCA).* IEEE. 523-526. doi:10.1109/ICDSCA53499.2021.9650091.

Peng, Qiaoyin. 2022. “Design of Intelligent Information Platform for Human Resource Allocation Based on Fuzzy Data Mining Algorithm.” *4th International Conference on Smart Systems and Inventive Technology.* IEEE. 1159-1162. doi:10.1109/ICSSIT53264.2022.9716414.

Pressman, Roger S., and R. Bruce Maxim. 2020. “The Software Process.” Chap. 2 in *Software engineering: A practitioners approach*, by Roger S. Pressman and R. Bruce Maxim, 25-36. McGraw Hill.

Royce, Winstion W. 1970. “Managing the Development of Large Software Systems: Concepts and Techniques.” *IEEE WESCON.* IEEE. 1-9.

Singhal, Palak. 2023. “Frontend vs Backend.” *GeeksforGeeks.* 06 January. Accessed January 21, 2023. https://www.geeksforgeeks.org/frontend-vs-backend/.

Sommerville, Ian. 2019. “Introduction to Software Engineering.” In *Software engineering, Tenth Edition*, by Ian Sommerville, 15-270. Pearson Education.

Tian, Fuzhang. 2022. “Human Resource Database Management Based on Apriori Algorithm.” *IEEE Asia-Pacific Conference on Image Processing.* IEEE. 1313-1316. doi:10.1109/IPEC54454.2022.9777457.

Zhang, Jinxiang, Qi Wang, Ying Pan, and Xu Liu. 2021. “Research on Archives Information Management System Based on Computer Big Data.” *IEEE International Conference on Power Electronics, Computer Applications (ICPECA).* IEEE. 729-732. doi:10.1109/ICPECA51329.2021.9362505.

# Appendix A: Software Requirements Specification

THE UNIVERSITY OF THE WEST INDIES

A picture containing diagram

Description automatically generatedST. AUGUSTINE, TRINIDAD & TOBAGO, WEST INDIES

FACULTY OF ENGINEERING

**Department of Electrical & Computer Engineering**

BSc. in Electrical & Computer Engineering

**ECNG 3020**

**Final Year Project**

**Software Requirements Specification**

MULTI-PURPOSE HUMAN RESOURCE INFORMATION SYSTEM

Avron Ramroop

# 1. Introduction

## 1.1 Purpose

The software requirements of the Human Resource Information System are specified in this document. This version introduces the web-based user interface for the system which is used together with a cloud-based database to facilitate the creation of HRIS accounts, the resetting of passwords, sending of request for information alerts to staff members, storing of human resource information and the generation of reports based on the stored human resource information.

## 1.2 Product Scope

The product is a web application for managing Human Resource Information for the members of staff at the faculty of Electrical and Computer Engineering at the University of the West Indies St. Augustine. The purpose of the system will be to store important human resource information of staff members to be accessed by the members of the administrative staff.



# 2. Overall Description

## 2.1 Product Perspective

This product is the first of its kind for the department of Electrical and Computer Engineering at the University of the West Indies, St. Augustine. It is designed to directly allow members of staff within the department to directly input and modify their essential human resource information such as bio data, Curriculum Vitae, Publication Achievements, awards etc. It allows administrative staff to modify staff accounts and generate reports based on the information stored for the staff members. The users of the system will be of two categories, staff user and administrator user.



## 2.2 Product Functions

The main functions of the product are outlined below:

1. Staff

1.1 Account Management

* Register Account
* Edit profile information
* Modify Password

1.2 Professional Achievements

* Add awards, publications and associated information

1.3 Information and Reports

* Request Information
* Generate Reports

1.4 View Contacts

2. Admin

2.1 Manage Accounts

* + - Add Account
    - Edit Password
    - View Staff List

2.2 Staff Reports

* Generate Staff Reports
* View Staff Information
* Request Staff Information

2.3 Information Requests

* View and Approve staff requests for information.

# External Interface Requirements

## User Interface (UI)

The user interface will be accessed via a person computer connected to the internet. It will allow users to register and log into the system and navigate through the system’s implemented functions.

## 3.2 Database

The system requires a database to store the required human resource information and allow queries to be made to view and edit the stored information. The database must be able to handle simultaneous queries and write functions for at least 100 users.

# Specific Requirements

* 1. Login/Register

4.1.1 Feature Description

This feature allows the user to log into their account and if one does not exist it allows them to create an account.

4.1.2 Stimulus/Response Sequences

User inputs email address and password

System verifies that the credentials are correct.

System redirects user to the homepage.

If user does not exist:

System redirects user to sign up page.

User inputs required information.

System verifies information.

System redirects user to the homepage.

4.1.3 Functional Requirements

|  |  |  |
| --- | --- | --- |
| Label | Requirement | Priority |
| REQ1-1 | The system will allow a staff user or admin user to be created | Necessary |
| REQ1-1.1 | The registration information includes two text fields for first name and last name, an email field and two fields for password and confirm password. | Necessary |
| REQ1-2 | The system will allow an existing user to log into the system | Necessary |
| REQ1-2.1 | The login information will include an email field and a password field | Necessary |

4.2 Update Profile Information

4.2.1 Feature Description

This feature will allow the user to view and edit essential profile information and upload/change a profile picture and a curriculum vitae document.

Profile Information

The required profile information:

* Title
* First Name
* Middle Name
* Last Name
* Gender
* Phone Number
* Bio
* Date of Birth
* Employee type
* Curriculum Vitae

Associated Information

The associated information to be stored:

* Publications
* Awards
* Conferences Attended
* Professional Development
* Manuscripts under review and in preparation
* Technical Presentations
* Research Grants
* Professional Consultancies
* University Public and Professional Service Roles
* Post Graduate Research and Supervision
* General Areas of Academic Specialization

4.2.2 Stimulus/Response Sequences

User inputs required information.

System verifies that information entered is valid within the fields.

System saves information and attachments to the database.

4.2.3 Functional Requirements

|  |  |  |
| --- | --- | --- |
| Label | Requirement | Priority |
| REQ2-1 | The system will display currently stored information and attachments associated with the staff member from the database | Necessary |
| REQ2-2 | The system will store new/edited information to the database | Necessary |
| REQ2-2.1 |  |  |

4.3 Request Information

4.3.1 Feature Description

This feature automatically sends staff users requests for missing information from their profile. These requests are sent in the form of automated emails to the staff users email address.

4.3.2 Stimulus/Response

Every 7 days the system will check the staff users profile information

The system sends an email to the staff user detailing the information missing from their profile.

4.3.3 Functional Requirements

|  |  |  |
| --- | --- | --- |
| Label | Requirement | Priority |
| REQ11-1 | The system will traverse each staff members profile and check for empty fields. An email will then be sent to the user urging them to update these missing fields. | Necessary |

4.4 Send Requests for Information to the Administrator

4.3.1 Feature Description  
User will be allowed to send requests for information to be processed by the administrators. Information will include:

* Study and Travel Grants
* Institutional Visit Allowance (IVA)/Development and Training Grant (DTG)
* Book Grant
* Claim Forms
* Pension plans

4.3.2 Stimulus/Response Sequences

User selects information to be requested

System stores request to be approved by administrator

4.3.3 Functional Requirements

|  |  |  |
| --- | --- | --- |
| Label | Requirement | Priority |
| REQ3-1 | The system will allow the user to select one or more articles of information. Information will include Study and Travel Grants, Institutional Visit Allowance (IVA)/Development and Training Grant (DTG), Book Grants, Claim Forms, Pension plans | Necessary |

4.5 View Contacts

4.5.1 Feature Description

This function allows the staff user to view important university contacts provided by the administrator.

4.5.2 Stimulus/Response

The user navigates to the contacts section.

The system displays the contacts to the user.

4.5.3 Functional Requirements

|  |  |  |
| --- | --- | --- |
| Label | Requirement | Priority |
| REQ5-1 | The system will display stored contacts to the user | Necessary |

4.6 Manage Accounts

4.6.1 Feature Description

This feature allows the administrator to view all staff accounts, add new accounts and reset existing accounts. It also allows the creation of new admin accounts.

4.6.2 Stimulus/Response

The user inputs the email address of an account to be created and the type of account (admin or staff)

The system verifies that a user does not already exist

The system creates the account.

The user selects an account to be reset

The system resets the account

4.6.3 Functional Requirements

|  |  |  |
| --- | --- | --- |
| Label | Requirement | Priority |
| REQ6-1 | The system will display all accounts to the user | Necessary |
| REQ6-1.1 | The system will allow the user to search for an account | Desirable |
| REQ6-1.2 | The system will allow the user to select one or more accounts to be reset | Necessary |
| REQ6-2 | The system will allow the user to add accounts by inputting a valid email address that does not already exist as an account. | Necessary |
| REQ6-2.1 | The system will allow the user to add multiple accounts by uploading a txt file with email addresses. | Necessary |

4.7 View Employees

4.7.1 Feature Description

This feature allows the administrator to view all staff members and their associated information.

4.7.2 Stimulus/Response

The user selects a staff member from the list.

The system displays the information associated with that staff member.

4.7.3 Functional Requirements

|  |  |  |
| --- | --- | --- |
| Label | Requirement | Priority |
| REQ7-1 | The system will display all staff members to the user | Necessary |
| REQ7-1.1 | The system will allow the user to search for a staff member | Desirable |
| REQ7-2 | The system will display all information input by the staff member. | Necessary |

4.8 Generate Reports

4.8.1 Feature Description

This feature allows the administrator to generate reports consisting of staff publications, technical presentations, awards, conferences attended honours or certificates obtained and contribution to the department or university by academic year. These reports may be downloaded in pdf format.

4.8.2 Stimulus/Response

The user enters the name of the staff member and the desired academic year.

The system displays the associated information..

4.8.3 Functional Requirements

|  |  |  |
| --- | --- | --- |
| Label | Requirement | Priority |
| REQ8-1 | The system allows the user to input the first and last names of the desired staff member along with the desired academic year. | Necessary |
| REQ8-1.1 | The system allows the user to download a copy of the requested information in pdf format | Necessary |

4.9 Update Contacts

4.9.1 Feature Description

This feature allows administrators to edit the list of important university contacts to be viewed by the staff users.

4.9.2 Stimulus/Response

User inputs contact information

Systems stores information to the database

User deletes contact information

System removes entry from the database.

4.9.3 Functional Requirements

|  |  |  |
| --- | --- | --- |
| Label | Requirement | Priority |
| REQ9-1 | The system will allow user to edit, add and delete contact information from the database. | Necessary |

# Appendix B: Software Design Description

THE UNIVERSITY OF THE WEST INDIES

A picture containing diagram

Description automatically generatedST. AUGUSTINE, TRINIDAD & TOBAGO, WEST INDIES

FACULTY OF ENGINEERING

**Department of Electrical & Computer Engineering**

BSc. in Electrical & Computer Engineering

**ECNG 3020**

**Final Year Project**

**Software Design Description**

MULTI-PURPOSE HUMAN RESOURCE INFORMATION SYSTEM

Avron Ramroop

1. Introduction
2. Goals and Objectives

The goals of the HRIS system are as follows:

* 1. Allows administrator to create or reset HRIS accounts, both manually and through a batch process.
  2. Allow staff to modify password after creation or reset of HRIS accounts.
  3. Sends all staff members Request for Information (RFI) alerts in the in the form of emails to stored email address. The nature of the requested information must be indicated in the email.
  4. Allows staff to directly update essential data to the system. The staff will be responsible for updating their profile. The essential data is as follows:
     1. Bio data including name, date of birth, contact number, email address, NIS number, PAYE number, etc.
     2. Updated Curriculum Vitae in .DOCX or PDF
     3. Publication Achievements with academic year - Journal papers, Conference Papers, Books
     4. Awards within academic year
     5. Fellowships, Memberships, Affiliations with academic year
  5. Allows managers to generate a variety of reports based on data entered by staff, eg. Annual Reports consisting of staff publication list, awards, fellowships, even documents required for staff evaluation, leave management, etc.
  6. Allows staff to request information from the system via request to the administrator, which will be displayed for viewing purposes. Information to be requested are as follows:
     1. Contract benefits including basic salary, allowances and all book and study grants entitled to.
     2. Staff training opportunities including short courses, certificate programmes, etc.
     3. List of all important university contacts for important operations such as applying for study leave, submission of book grant claims, etc.

1. Scope

The product is a web application for managing Human Resource Information for the members of staff at the faculty of Electrical and Computer Engineering at the University of the West Indies St. Augustine. The purpose of the system will be to store important human resource information of staff members to be accessed by the members of the administrative staff.

1. System Overview

The following diagram shows a high-level overview of the system.



1. Definitions and Abbreviations

Admin – Administrator

DFD – Data Flow Diagram

HRIS – Human Resource Information System

UI - User Interface

1. References

Foundation, Django Software. 2022. “Django Documentation.” *Django.* 3 August. Accessed January 9, 2023. <https://docs.djangoproject.com/en/4.1/>.

Foundation, Python Software. 2023. *Python 3.10.10 documentation.* 10 February. Accessed February 10, 2023. <https://docs.python.org/3.10/index.html>.

1. System Architecture

The main system of the HRIS is divided into three main modules for its two types of users. These are the admin module and the staff module and the authentication module. Each of these modules contains their own sub modules as shown below.



1. Authentication Module

The authentication module handles the login and logout of users. It also allows users to change their password if so desired. Upon verification, the user is given access to either the staff module or the admin module depending on the type of the user.

1. Admin Module

The admin module serves as the main module to be used by the administrator user. It is comprised of several sub modules that these users have access to.

|  |  |
| --- | --- |
| **Sub Module** | **Description** |
| Employees Sub Module | This module allows admin users to view all employees and all the information stored in the database associated with each employee. This information includes:   * Title * First Name * Middle Name * Last Name * Gender * Phone Number * Bio * Date of Birth * Employee type * Curriculum Vitae * Publications * Awards * Conferences Attended * Professional Development * Manuscripts under review and in preparation * Technical Presentations * Research Grants * Professional Consultancies * University Public and Professional Service Roles * Post Graduate Research and Supervision * General Areas of Academic Specialization |
| Reports Sub Module | The reports submodule of the admin module allows admin users to generate reports of the publications, awards, technical presentations, and conferences attended by a specific employee based on the academic year of the aforementioned achievements. The admin user can then generate a pdf file documenting these results. |
| Manage Accounts Sub Module | This sub module allows admin users to view all staff accounts and reset or delete them. The admin user can also add new accounts to the system. |
| Contacts Sub Module | This sub module allows the admin user to edit the page of important university contacts. The user can add or remove contacts to be viewed by the staff user. |
| Requests Sub Module | The requests submodule of the admin module allows admin users to view requests for information submitted by staff users. |

1. Staff Module

The staff module serves as the main module to be used by the staff user. It is comprised of several sub modules that these users have access to.

|  |  |
| --- | --- |
| **Sub Module** | **Description** |
| Profile Sub Module | The profile sub module allows staff users to view and edit their individual profile information. (Information outlined in the employees sub module of the admin module) |
| Leave Sub Module | The leave sub module of the staff modules allows the staff user to submit a leave request by inputting the start date, end data, leave type and reason for leave. |
| Requests Sub Module | The requests sub module of the staff module allows a staff user to submit a request for information to be reviewed by the admin user. |
| Contacts Sub Module | The contacts sub module of the staff module allows staff user to view the contact information supplied by the admin users. |

1. Level 1 DFD for Authentication Module



1. Level 2 DFD for Authentication Module



1. Level 2 DFD for Admin Module



1. Level 2 DFD for Staff Module



1. Data Design

In order to store and manage the data required by the system, several database objects must be created. These objects are known as tables. The following section shows the tables required by this system.

Table 34: User Information

|  |  |  |
| --- | --- | --- |
| Column Name | Data Type | Description |
| username | Varchar | Email address |
| password | Varchar | Password of the user |
| user\_type | Varchar | Admin or Staff |

Table 35: Employee Information

|  |  |  |
| --- | --- | --- |
| Column Name | Data Type | Description |
| firstname | Varchar | First Name |
| lastname | Varchar | Surname |
| othername | Varchar | Other Name |
| sex | Varchar | Gender |
| email | Varchar | Email |
| tel | Int | Phone Number |
| bio | Varchar | Biography |
| employeetype | Varchar | Fulltime, Part-time, Contract, Intern |
| nisnumber | Int | NIS Number |
| birthday | date | Date of Birth |
| employeeid | Int | Employee ID Number |
| vitae | file | Curriculum Vitae Document |
| image | file | Profile Image |

Table 36:Publications

|  |  |  |
| --- | --- | --- |
| Column Name | Data Type | Description |
| title | Varchar | Title of Publication |
| publicaitontype | Varchar | Peer Reviewed Journal, Book, Conference Paper |
| year | Varchar | Academic Year |

Table 37: Awards

|  |  |  |
| --- | --- | --- |
| Column Name | Data Type | Description |
| title | Varchar | Title of Award |
| year | Varchar | Academic Year |

Table 38: Unpublished Manuscripts

|  |  |  |
| --- | --- | --- |
| Column Name | Data Type | Description |
| title | Varchar | Title of Manuscript |
| status | Varchar | In Preparation or Under review |

Table 39: Honours and Certificates

|  |  |  |
| --- | --- | --- |
| Column Name | Data Type | Description |
| title | Varchar | Title of Honour/Certificate |
| competition | Varchar | Competition honour was obtained |
| year | Varchar | Academic Year |

Table 40: Areas of Academic Specialisation

|  |  |  |
| --- | --- | --- |
| Column Name | Data Type | Description |
| area | Varchar | Area of Specialisation |

Table 41: Research Grants

|  |  |  |
| --- | --- | --- |
| Column Name | Data Type | Description |
| title | Varchar | Title of Research Grant |

Table 42: Post Graduate Research and Supervision

|  |  |  |
| --- | --- | --- |
| Column Name | Data Type | Description |
| title | Varchar | Title of Thesis/Project |
| firstname | Varchar | First name of Student Supervised |
| lastname | Varchar | Surname of Student Supervised |
| level | Int | Degree Level of Student |
| year | Varchar | Academic Year |

Table 43: Research Interests and Associations

|  |  |  |
| --- | --- | --- |
| Column Name | Data Type | Description |
| research | Varchar | Name of research field |
| interest | Varchar | interest |

Table 44: Conferences Attended

|  |  |  |
| --- | --- | --- |
| Column Name | Data Type | Description |
| title | Varchar | Name of Conference |
| year | Varchar | Academic Year |

Table 45: Technical Presentations

|  |  |  |
| --- | --- | --- |
| Column Name | Data Type | Description |
| title | Varchar | Title of Presentation |
| year | Varchar | Academic Year |

Table 46: University Professional Service Roles

|  |  |  |
| --- | --- | --- |
| Column Name | Data Type | Description |
| role | Varchar | Title of the Role |
| association | Varchar | Association of the Role |
| date | date | Date |

Table 47: Professional Consultancies

|  |  |  |
| --- | --- | --- |
| Column Name | Data Type | Description |
| title | Varchar | Title of Consultancy |
| position | Varchar | Position |
| period | Varchar | Time period |

Table 48: Professional Development

|  |  |  |
| --- | --- | --- |
| Column Name | Data Type | Description |
| title | Varchar | Title |
| year\_start | Int | Year Commenced |
| year\_end | Int | Year Ended |

Table 49: Contributions to the Department, Faculty or University

|  |  |  |
| --- | --- | --- |
| Column Name | Data Type | Description |
| contribution | Varchar | Contribution |
| year | Varchar | Academic Year |

Table 50: Other Professional Activities

|  |  |  |
| --- | --- | --- |
| Column Name | Data Type | Description |
| activity | Varchar | Other Activity |

Table 51: Request to the Administrator

|  |  |  |
| --- | --- | --- |
| Column Name | Data Type | Description |
| information | Varchar | Requested information |
| message | Varchar | Reason for Request |

Table 52: Contacts

|  |  |  |
| --- | --- | --- |
| Column Name | Data Type | Description |
| information | Varchar | Information |
| email | Varchar | Email Address |
| name | Varchar | Name of Contact |

1. Data Description

A picture containing diagram

Description automatically generated

1. User Interface Design

eGraphical user interface

Description automatically generated

Figure 28: Login Page

Graphical user interface

Description automatically generated

Figure 29: Registration Page

Graphical user interface, website

Description automatically generated

Figure 30: Staff Home Page

Graphical user interface, application

Description automatically generated

Figure 31: Staff Profile Page

A picture containing application

Description automatically generated

Figure 32: Staff Profile Page Continued

Graphical user interface, text, application, email

Description automatically generated

Figure 33: Staff View Contacts

Graphical user interface, application

Description automatically generated

Figure 34: Staff Submit Request For Information Page

Graphical user interface, application, website

Description automatically generated

Figure 35: Admin Home Page

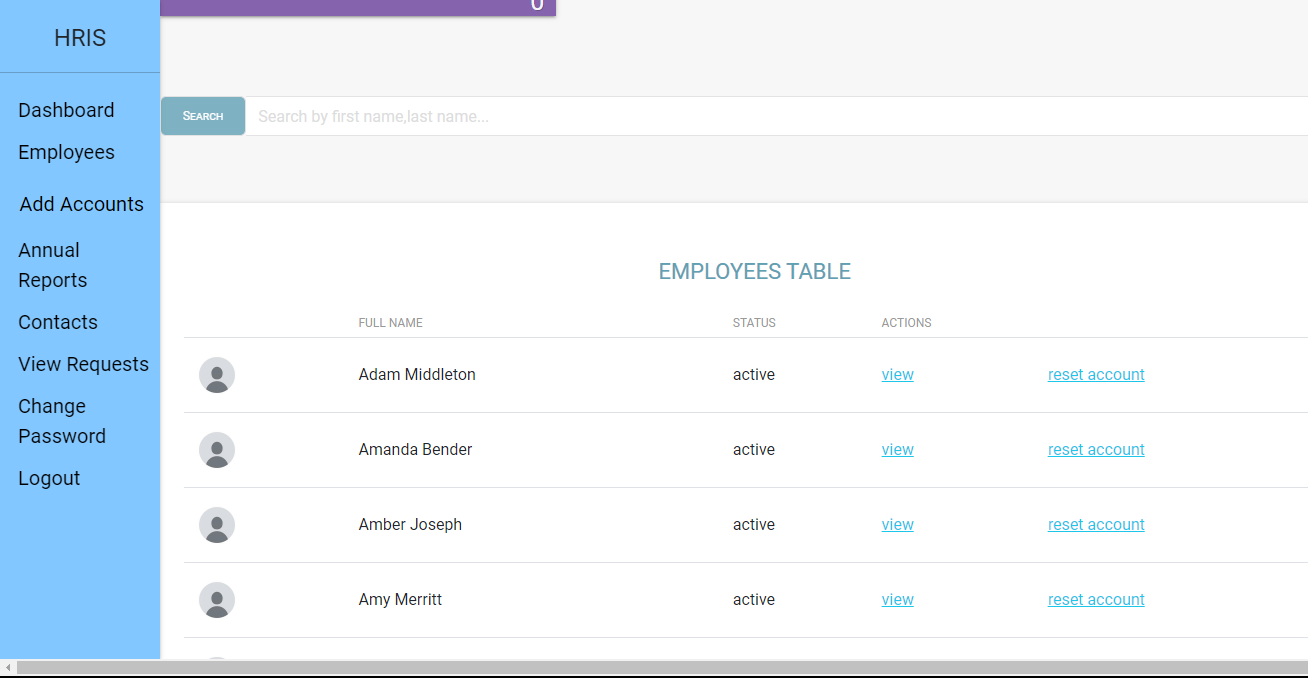


Figure 36: Admin View Employees Page

Graphical user interface, application

Description automatically generated

Figure 37: Admin Add Account Page

Graphical user interface, text, application, email

Description automatically generated

Figure 38: Admin Batch Add Accounts Page

Graphical user interface, text, application, email

Description automatically generated

Figure 39: Admin View Contacts Page

Graphical user interface, application

Description automatically generated

Figure 40: Admin Annual Reports Page 1/3

Graphical user interface, text, application, email

Description automatically generated

Figure 41: Admin Annual Reports Page 2/3

Graphical user interface, text, application, email

Description automatically generated

Figure 42: Admin Annual Reports Page 3/3

1. Testing

## URL Testing

The Django TestCase library was used to test all the URLs defined within the system to ensure that the correct view functions were called as a result of the URL. In total 76 URL tests were performed and all were passed.

## View Function Testing

Registration Function

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Test Case | Steps | Expected Results | Status |
| 1 | Successful Registration | Input first name, last name, valid email address, password, matching confirmation password, | CustomUser object created.  Employee object created.  StaffUser object created.  User is redirected to login page. | Pass |
| 2 | Missing Information | Leave one or more fields blank, | Error message displayed to the user. | Pass |
| 3 | Password Mismatch | Input first name, last name, valid email address, password, non- matching confirmation password, | Error message displayed to the user. | Pass |

Login Function

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Test Case | Steps | Expected Results | Status |
| 1 | Valid Login | Input valid login credentials | User is redirected to either admin or staff homepage depending on the user type. | Pass |
| 2 | Invalid Login | Input invalid login credentials | Error message is displayed to the user. | Pass |

Add Account Function

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Test Case | Steps | Expected Results | Status |
| 1 | Add Account Authenticated | Attempt to access function while logged in with an admin account. | Add Account Page rendered to user. | Pass |
| 2 | Add Account Unauthenticated | Access function while not logged in | User redirected to login page. | Pass |
| 3 | Duplicate Email | Access function while logged in with an admin account.  Enter email address that is already in use | Error displayed and user is redirected to the Add Account page. | Pass |
| 4 | Invalid Email | Access function while logged in with an admin account.  Enter email address that does not end with “sta.uwi.edu” | Error displayed and user is redirected to the Add Account page. | Pass |

Batch Add Accounts Function

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Test Case | Steps | Expected Results | Status |
| 1 | Txt file Input | Access function while logged in with an admin account.  Upload .txt file containing valid emails | CustomUser, StaffUser and Employee objects are created for each email in file and a success message is displayed to the user. | Pass |
| 2 | Non Txt file Input | Access function while logged in with an admin account.  Upload a non .txt file | Error displayed and user is redirected to the Batch Add Account page. | Pass |
| 3 | Txt file contains Invalid emails | Access function while logged in with an admin account.  Upload .txt file containing invalid emails | Error displayed showing all invalid email addresses and user is redirected to the Batch Add Account page. | Pass |

View Model Data Function

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Test Case | Steps | Expected results | Status |
| 1 | Authenticated user | Access function while logged in | View page is rendered to the user. | Pass |
| 2 | Unauthenticated user | Access function while not logged in | User is redirected to the login page. | Pass |

Add Model Object Functions

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Test Case | Steps | Expected Results | Status |
| 1 | Valid information | Access function while logged in.  Submit valid information into the form fields. | New object added to model with submitted information.  Success message displayed to user. | Pass |
| 2 | Invalid Information | Access function while logged in.  Submit invalid information into the form fields. | Error displayed to user.  User is redirected to Add object page. | Pass |

Edit Model Object Functions

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Test Case | Steps | Expected Results | Status |
| 1 | Valid object id | Access function while logged in.  Submit request with valid object id to be edited | Functions retrieves object with id equal to the requested id and user is allowed to edit object. | Pass |
| 2 | Invalid object id | Access function while logged in.  Submit request with invalid object id to be edited | 404 error displayed to user. | Pass |

Delete Model Object Functions

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Test Case | Steps | Expected Results | Status |
| 1 | Valid object id | Access function while logged in.  Submit request with valid object id to be deleted | Functions retrieves object with id equal to the requested id and deletes object from database. | Pass |
| 2 | Invalid object id | Access function while logged in.  Submit request with invalid object id to be deleted | 404 error displayed to user. | Pass |

View All Employees Function

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Test Case | Steps | Expected Results | Status |
| 1 | Authenticated admin user | Access function while logged in to admin account | View Employees page is displayed to the user | Pass |
| 2 | Unauthenticated user | Access function while not logged in | User is redirected to the login page. | Pass |

View Employee Function

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Test Case | Steps | Expected Results | Status |
| 1 | Authenticated admin user | Access function while logged in to admin account | View Employees page is displayed to the user | Pass |
| 2 | Unauthenticated user | Access function while not logged in | User is redirected to the login page. | Pass |

Annual Reports Function

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Test Case | Steps | Expected Results | Status |
| 1 | Valid first name and last name | Access function while logged in to admin account.  Submit valid employee first name and last name to function | Report with employee information for given academic year is rendered to the user. | Pass |
| 2 | Invalid first or last name | Access function while logged in to admin account.  Submit invalid employee first name and last name to function | Error is displayed to the user. The user is redirected to the Annual Reports page. | Pass |

# Appendix C: Code Snippets

Text

Description automatically generated

Snippet 1: CustonUser Model

Text

Description automatically generated

Snippet 2: AdminUser and StaffUser Models

Text

Description automatically generated

Snippet 3: Registration Function

Text

Description automatically generated

Snippet 4: Login Function

Text

Description automatically generated

Snippet 5: Add Account Function

Text

Description automatically generated

Snippet 6: Batch Add Staff Accounts Function

Text

Description automatically generated

Snippet 7: Reset Staff Account Function

Text

Description automatically generated

Snippet 8: Awards Model

Text

Description automatically generated

Snippet 9: Publications Create Form

Text

Description automatically generated

Snippet 10: View Publications Function

Text

Description automatically generated

Snippet 11: Add Publication Function

Text

Description automatically generated

Snippet 12: Edit Publication Function

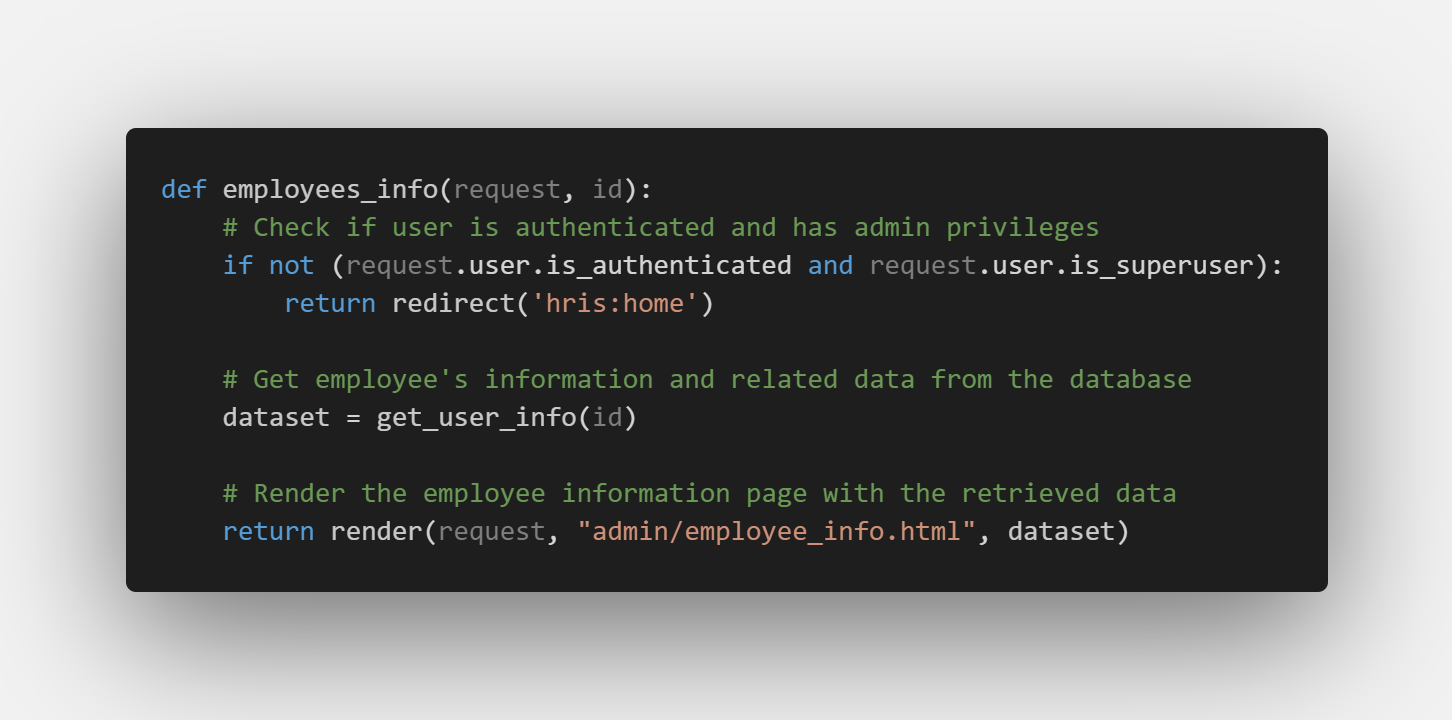
Text

Description automatically generated

Snippet 13: Delete Publication Function



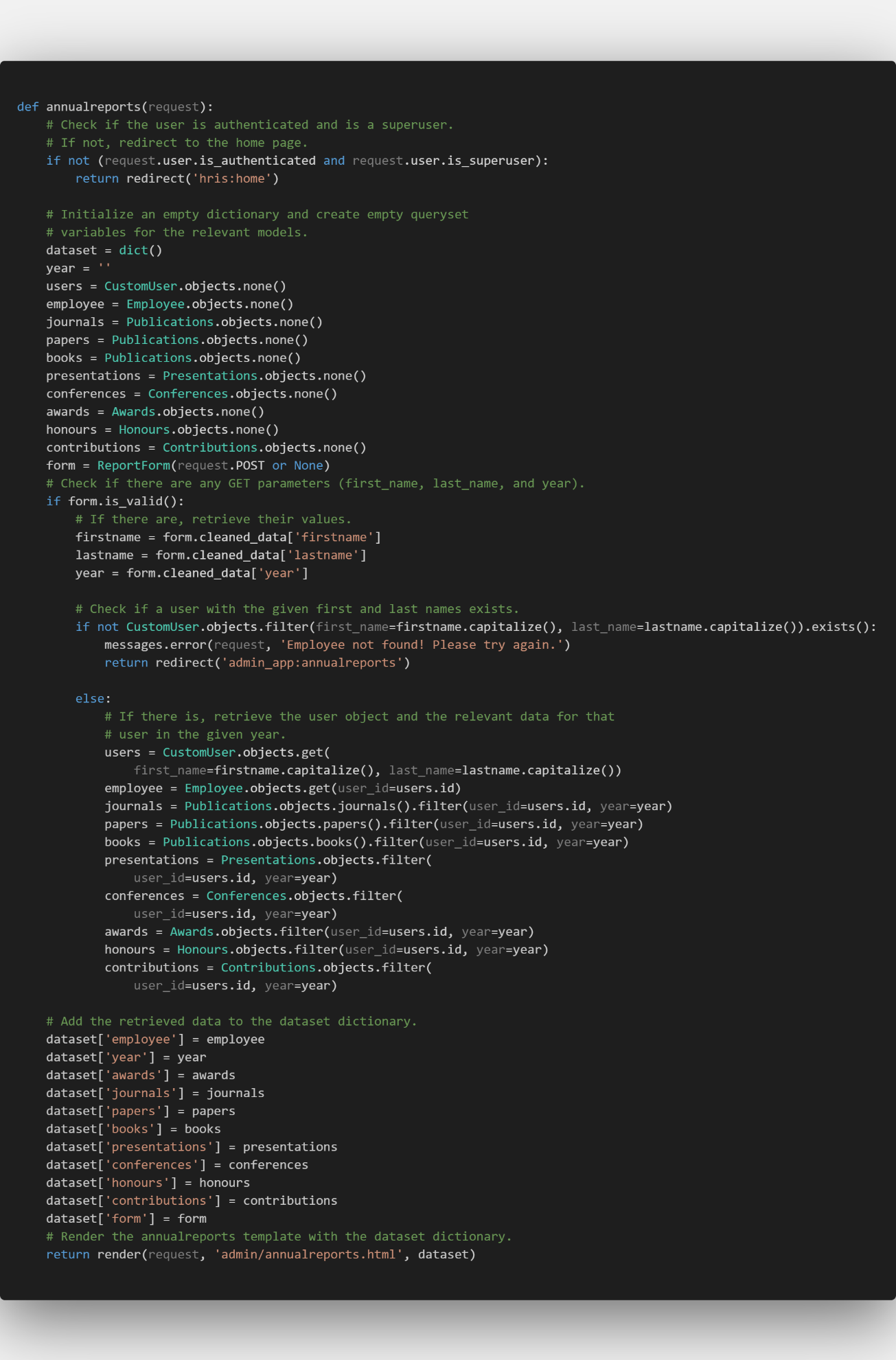
Snippet 14: View all Employees Function



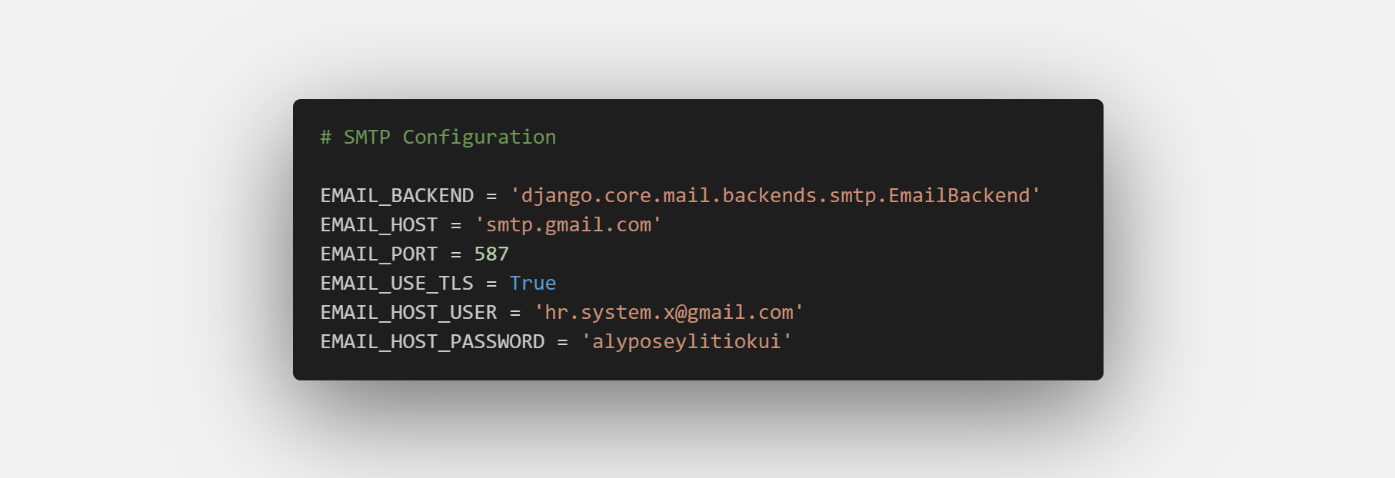
Snippet 15: Employee Information Function



Snippet 16: Get User Info Function



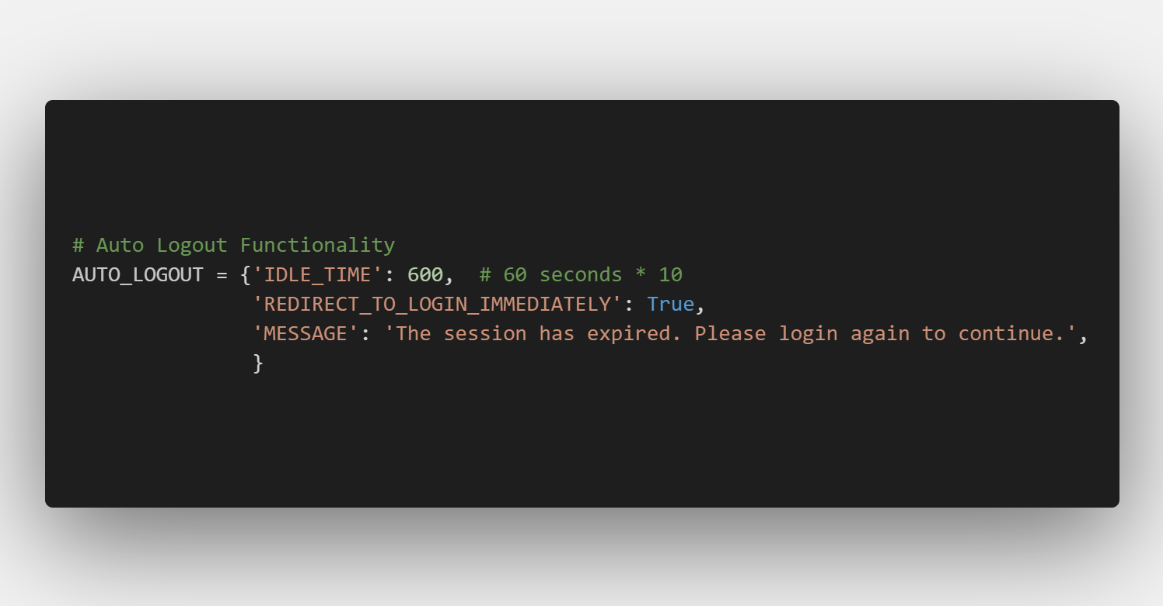
Snippet 17: Annual Reports Function



Snippet 18: SMTP Configuration



Snippet 19: Partial Code of the Automatic Requests for Information Function



Snippet 20: Auto Log Out Configurations