Author: Samuel Gilbert

Table of Contents

2 Analysis 3

2.1 Background 3

2.2 Description of current system 4

2.2.1 Interview with Ann – the absence administrator for the company 4

2.3 Problem identification 7

2.3.1 Questionnaire to solicit feedback from current users of the system 7

2.3.2 Reponses 7

2.4 User identification 8

2.5 User needs and limitations 9

2.6 data sources and destinations 10

2.7 Data volumes 11

2.8 analysis data dictionary 11

2.9 Data flows 12

2.9.1 Existing data flows 12

2.9.2 Proposed data flows 13

2.10 Objectives 14

2.11 complexity 15

2.12 potential solutions 16

2.13 use of formal methods 17

2.14 entity relationship model 18

3 Design 19

3.1 Overall system design 19

3.2 Modular structure 20

3.3 Database structure AND VALIDATION 21

3.3.1 Public Holiday Table 21

3.3.2 Date Table 22

3.3.3 Approved Absence Booking Date Table 22

3.3.4 Approved Absence Booking Table 23

3.3.5 Absence Type Table 24

3.3.6 Employee Table 25

3.3.7 Ad Hoc Absence Request Table 26

3.3.8 Main Vacation Request Table 27

3.3.9 Company Role Table 28

3.4 file organisation and processing 29

3.5 database design and entity relationship model 29

3.6 storage media and format 31

3.7 algorithms DESIGN 32

3.7.1 Annual Leave required for a booking Algorithm 32

3.7.2 Annual Leave Remaining Algorithm 33

3.7.3 Has Sufficient Annual Leave Algorithm 34

3.7.4 Are Sufficient Staff in Role to Grant Request Algorithm 35

3.7.5 Process Absence Request Algorithm 36

3.7.6 Main Vacation Request Allocation Algorithm 38

3.7.7 Ad Hoc Request Processing Algorithm 39

3.8 user interface 40

3.8.1 Login Page 41

3.8.2 Member of Staffs Homepage 41

3.8.3 Office Managers Homepage 43

3.8.4 Administrators Homepage 44

3.9 Security and integrity of data 47

3.9.1 Data Security 47

3.9.2 Data Integrity 47

3.9.3 System Security 48

3.10 test strategy 49

3.10.1 Introduction 49

3.10.2 Top Down Based Approach 49

3.10.3 Trace Tables 49

3.10.4 Unit Testing 49

3.10.5 System and Integration Testing 49

3.10.6 Security Testing 49

4 technical solution 51

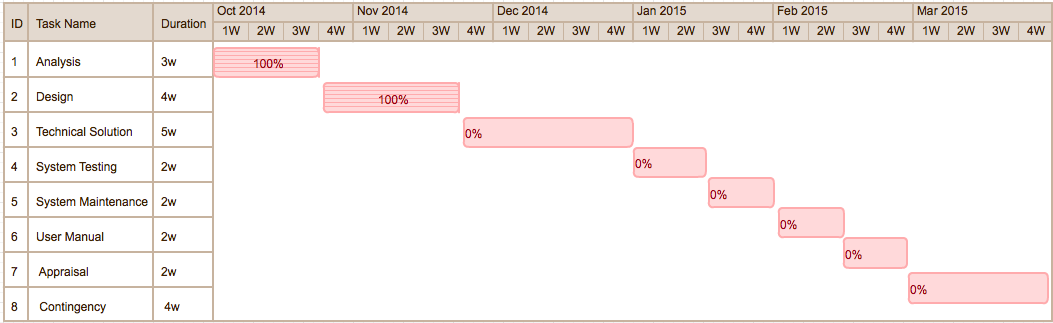
5 system testing 52

6 system maintenance 52

7 USER manual 52

8 appraisal 52

**WORK PLAN**



# Analysis

## Background

A local company, who does not wish to be named, needs to ensure that a minimum staffing level for each function within the business is maintained when reviewing and granting staff leave and other absence requests.

A member of the company, Ann, performs the role of absence administrator. Ann is a friend of my family and during a social event earlier this year I was discussing my A level work with her and the discussion turned to my Computing course and project.

During this discussion, Ann highlighted the current manual processes involved in administering the company’s’ absence requests and some of the problems with these processes. She asked if I would be able to create a computer system that would help to automate these manual procedures, simplifying and speeding up the current process.

This is the basis of my COMP 4 Project, as detailed in this report.

## Description of current system

To better understand the current system I spent an hour interviewing the person who performs the role of absence administrator. Q&A from this interview is detailed below.

### Interview with Ann – the absence administrator for the company

Q. Describe the current process that is used for handling staff absence requests.

A. There are two main parts to the process.   
  
Firstly, between October and November each year we ask all staff to write down their first and second choices for their main two-week vacation in the upcoming year. Once everyone has done this, it is my task to try and accommodate as many people’s requests as possible whilst ensuring that we keep to the minimum staffing levels needed for each role within the company.

Secondly, once everyone’s main two weeks leave have been allocated, for the rest of the year it is my job to handle ad-hoc absence requests from staff. My role is to check to see whether each absence request would take us below the minimum staffing level for the role that this person performs. If we are above the minimum staffing levels for the absence period requested, then the request is approved. If the request would take us below the minimum staffing level then the request is denied or passed to the office manager.

Q. What mechanism do you use to track who is absent and when?

A. We have a large paper based wall chart, which we record approved absences on.

Q. For the main two week vacation allocation in Oct / Nov, you mentioned that you try to   
 accommodate as many requests as possible. Could you describe how you do this?

A. All staff members write down the dates for their first and second choice for their main two weeks leave on a form and give this to me. I then sort these requests into an order based on each member of staff’s length of service with the company. The longest serving member of staff is top of the pile; the shortest serving member of staff is bottom of the pile.

I then go through the forms one at a time.

For each form I look at the first choice and check the wall chart to see how many people in the same role already have that time booked off. If the number of staff in the office during that period for that role is above the minimum needed, then I grant the first choice and record it on the chart.

If not, I repeat this check for the second choice. If the second choice can be granted I update the chart, otherwise I stop further work on the forms and go back to the person to ask them to give me another choice. I do not process any more forms until I have an agreed date for that person.

Q. Does everyone have to book a main two weeks? What if someone only wants to take one week   
 off? What if someone wants to take three weeks off?

A. Generally we ask everyone to select a main two weeks. If someone only wants to select a week they can, but their second week will have to be booked as part of the ad-hoc process once everyone else’s main two weeks have been granted.

Three-week vacations are discouraged, but sometimes do occur (e.g.: Long vacations for Weddings / Honeymoons etc.). Anyone requesting a longer leave period must first gain approval from the office manager prior to submitting his or her request.

Q. Can you describe how the ad-hoc process for requesting absence differs from what you’ve   
 described already?

A. After the main two weeks have been agreed for everyone (in October / November) we allow people to request any other absence they require throughout the year. People do this by filling in a simple form stating the dates they wish to book and then they place in my in-tray. I tend to process these requests once a week, although people sometimes ask me to process their request immediately if it is urgent.

The process is similar to the one for the main two weeks, although I process the request forms in the order I receive them (first submitted, first processed). Basically, for each form I check the date requested and if sufficient staff are available to cover the minimum level for the role, then I grant the request and add to the wall-chart, otherwise I reject. Either way, I let the person who requested the leave know either by email or by telling them directly.

It’s a bit different for non-annual leave absence requests?

Q. What other types of absence requests are there? How do they differ from annual leave   
 requests?

A. Annual leave forms the majority of requests for absence, but there are others, namely:

* Training Requests  
  Sometimes members of staff need to attend training courses. Such requests are handled in the same way as ad-hoc annual leave requests, IE: Only approving if minimum staffing levels for that persons role are maintained. The only difference between training requests and annual leave requests is that training days do not consume peoples annual leave entitlement.
* Sickness

Generally sickness is not something planned in advance. Usually it is only known about when a member of staff phones in to say they are sick and will not be at work. Sometimes sickness time is known in advance, for example if a person is undergoing an operation and has a date for when this will take place. We use the wall-chart to record absence due to sickness but unlike the annual leave and training, there is no approval/rejection of sickness.

Sickness can leave the company under resources for a given role. When recording sickness, if the number of people for a role will be below the minimum required I inform the office manager, who will decide what to do (usually hiring temporary staff to cover the shortfall).

Absence due to sickness does not use peoples annual leave entitlement. However, we do keep records of how many days each member of staff has taken sick each year.

* Compassionate Leave

Compassionate Leave is granted in the event of a distressing personal circumstance, typically a death of a close family member. A member of staff would inform the office manager of this, and the office manager will inform me.

In terms of process, this is identical to sickness, IE: it is not subject to approval / rejection and can leave us below the minimum resourcing level for a role, in which case I inform the office manager who will determine what to do.

Compassionate leave does not use peoples annual leave entitlement, but we do track how many days a person has taken due to compassionate leave each year.

* Emergency Leave

Emergency Leave refers to annual leave that has to be taken on the same day due to an emergency. An example would be if a member of staff had been burgled or had a burst pipe or childcare issues. The member of staff would phone the office manager to inform them they will be absent and the office manager will provide me with the details. As with Sickness and Compassionate leave, this may leave us below the minimum resourcing level for a role in which case I inform the office manager who decides what to do. Emergency leave does use up days from a persons annual leave entitlement.

Q. How many ad-hoc requests do you have to process each week?

A. It varies, but probably about 3 or 4 requests would be average.

Q. Can you describe what IT equipment the staff have access to?

A. Everyone has either a desktop or a laptop computer. We all have a company email account. There are laser printers in the office.

Q. What about internet access?

A. Yes. We have an internal intranet, but everyone also has access to the internet.

Q. You mentioned there are different roles in the company, is there a list?

A. Yes, but really just a list on my desk. The roles are Manager, Customer Advisor and Cashier. There needs to be a minimum of 1 manager, 2 customer advisors and 3 cashiers.

Q. How many people work in the company?

A. We have a manager, deputy manager and supervisor (all are considered to be manager roles), 4 customer advisors and 5 cashiers. I am the supervisor, and part of my job is absence administration.

Q. How many days annual leave do people get?

A. Everyone gets 25 days a year annual leave, plus of course public holidays and weekends

Q. What would you say was an average number of total days absence per year for each person?

A. It really does depend, as some absences can’t be foreseen, but as a rough figure I’d say:  
  
Annual Leave - including Emergency Leave ( 25 days )  
Sickness (7 days)  
Compassionate Leave (most people would not use this in a year, but if taken up to 5 days)

Q. So the system doesn’t cover bank holidays / weekends?

A. The office is not open at weekends or public holidays. Those are non working. No minimum staffing levels, and non working days that occur during a holiday period do not count as annual leave.

Q. Would you be ok for me to produce a short survey asking for feedback on the current system? Would you be able to complete this, and also get the office manager and some of the members of staff to complete this?

A. Yes, I’d be happy to.

## Problem identification

### Questionnaire to solicit feedback from current users of the system

The following text was sent to Ann, who then emailed these questions to the staff and office manager. Ann consolidated the responses and sent to me, also adding her own responses to the questions.

“*As you know, the current procedures for booking and tracking absence are quite manual and we are currently considering whether there may be an opportunity to make more use of technology in helping to improve these processes.*

*As part of the analysis of this, we would welcome your views based on a short set of questions below. Please could you email me back with your responses by the end of next week (Friday 12th September).*

*Q1. What do you like about the current process?*

*Q2. What do you dislike about the current process?*

*Q3. List the top two things you would like the new system to improve.”*

### Reponses

A consolidated set of response to these questions is given below:

*Q1. What do you like about the current process?*

* *The way in which we grant leave seems fair.*
* *Ann is always helpful.*
* *The wall-chart let’s me see whose in and whose off when.*

*Q2. What do you dislike about the current process?*

* *Sometimes it can be a bit frustrating waiting for approval for leave.*
* *When Ann is off, leave requests go unanswered.*
* *I keep my own record of what leave I’ve booked as it’s hard to find that on the wall chart.*
* *Sometimes the wall-chart is not up to date.*
* *Doing the absence administration takes up too much of my time. I have other work to do.*

*Q3. List the top two things you would like the new system to improve.”*

* *It’d be great if I could get an instant response to holiday requests.*
* *I’d like to be able to see up to date information on what leave I’ve booked and how many days leave I have left to take.*
* *As the office manager, I’d like to see an up to date view on who is off at any given time and the resource levels for each role within the company.*
* *The new system should be much faster than the current process.*
* *As the holiday administrator, I’d like the system to automatically perform the processing for the main two week allocations, as this takes me a lot of time.*
* *Whatever system we have, it needs to be up to date and accurate.*
* *I’d like to be able to get information from my PC, not have to pester Ann all the time.*

## User identification

There are three main types of user.

* Staff member

Everyone working within the company is a staff member. Staff members will request absences for a number of reasons (eg: Annual Leave, Training, Sickness, and Compassionate Leave).

* Absence Administrator

The absence administrator is the person who administers the absence process, reviewing, granting or rejecting absence requests and ensuring that the minimum level of resourcing for each function within the company is maintained.

* Office Manager

The Office manager is responsible for the overall running of the office and needs visibility of which staff are absent at any given time.

## User needs and limitations

Each type of user has specific needs and limitations.

|  |  |  |
| --- | --- | --- |
| Type of user | Needs | Limitations |
| Members of Staff | Each member of staff needs to be able to create, update or cancel their own requests for absence and to see all of their requests.  Each member of staff needs to know their total number of days annual leave. This should be broken down into the total annual leave entitlement, the number of days already booked and the number of days remaining.  Members of staff should have visibility of how many staff of a given role are absent on any given day. | Should only be able to create, update, cancel absence requests or view information relating to their own absences. A member of staff must not be able to create, alter, view or remove any absence information for other members of staff. |
| Absence Administrator | To add new staff to the absence system when they join the branch.  To remove staff from the absence system if they leave.  To update staff information.  To define a list of roles within the branch.  To record the role(s) that each member of staff can perform.  To record information on the minimum number of staff needed in each role at any given time.  Once a year, to perform the initial allocation of each member of staffs main two weeks leave, ensuring that minimum staffing levels for each role are met.  On an ad-hoc basis, review and approve or reject other absence requests from staff.  To be able to create absence requests on behalf of a member of staff. EG: If a member of staff is taken ill or phones in sick. |  |
| Office Manager | Office Managers need to be able to see all absence requests for all staff members.  Office managers need to be able to view all staff on leave for any given week. | Office managers should not be able to create, update, cancel or delete absence information. This is the role of the Absence administrator. |

## data sources and destinations

|  |  |  |
| --- | --- | --- |
| Data | Source | destination |
| Main Two Week Vacation Request  - Name of Requestor   * First Choice Start Date * First Choice End Date * Second Choice Start Date * Second Choice End Date | Member of Staff (During Oct/Nov) | Holiday Administrator |
| Main Two Week Vacation Approval   * Name of Requestor * Approved Start Date * Approved End Date | Holiday Administrator | Member of Staff |
| Ad Hoc Absence Request  - Name of Requestor   * Start Date * End Date * Type of Request (Annual Leave, Training,Sickness,Compasionate Leave, Emergency Leave) | Member of Staff | Holiday Administrator |
| Ad Hoc Absenc Approval   * Name of Requestor * Approved Start Date   - Approved End Date | Holiday Administrator | Member of Staff |
| Recorded Absences | Holiday Administrator (from requests) | Wall Chart |
| List of who is absent when | Wall Chart | Office Manager Holiday Administrator |
| List of Public Holidays | The Internet | Wall Chart |

## Data volumes

Number of staff in the office: 12

Average number of days absence: 30 to 40.

Total individual days absence per year for the office that need to be recorded:

12 x 40 = 480 absence days needed to be recorded per year.

During October / November, every member of staff has to provide two choices for their main two weeks of vacation.

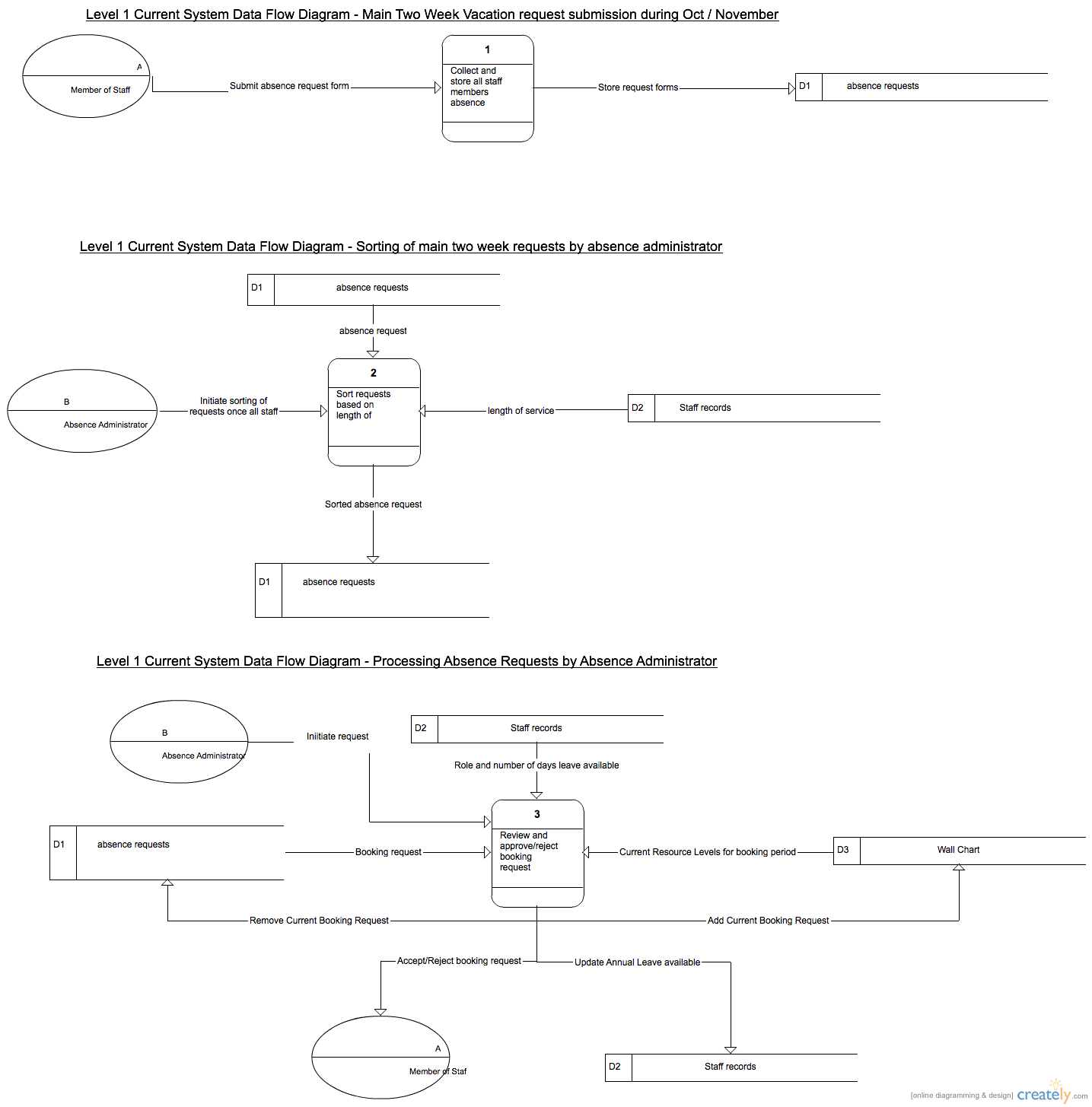
During the rest of the year, the holiday administrator processes on average 3-4 absence requests per week.

## analysis data dictionary

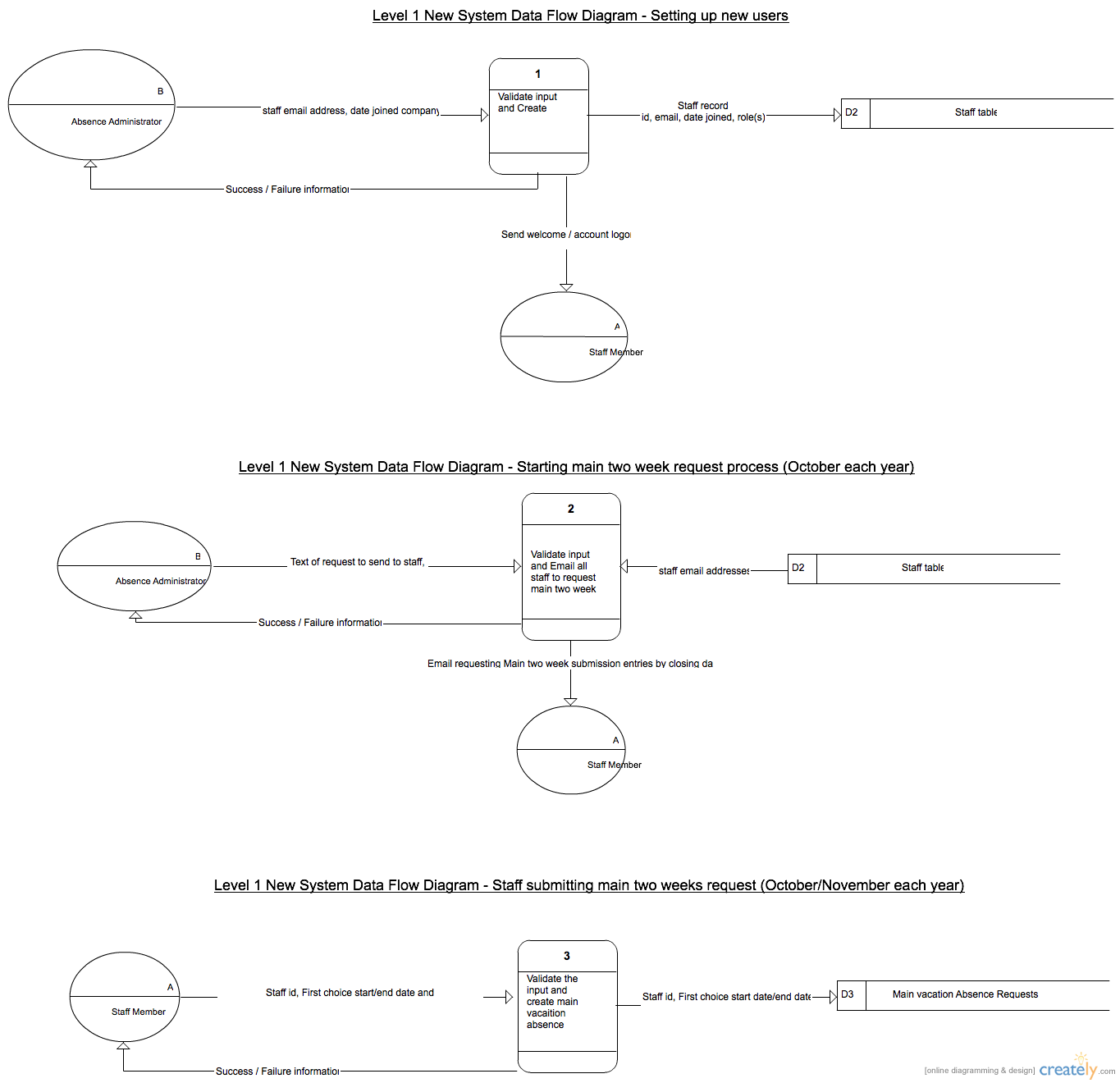
|  |  |
| --- | --- |
| Field | description |
| Staff Members Name | Name of the member of staff |
| Staff Members Role(s) | List of roles that the member of staff performs.  Roles are:  Manager, Customer Advisor, Cashier. |
| Staff Members Annual Leave Entitlement | How many days annual leave the member of staff is entitled to per year. |
| Absence Start Date | The date on which an absence commences. |
| Absence End Date | The date on which an absence ends. |
| Absence Type | The type of absence. This can be annual leave, sick leave, compassionate leave, training leave or emergency leave. |
| Absence Status | The status of an absence request. A request can be: Pending – means that it has not yet been approved or denied.  Approved – means that the request has been approved by the absence administrator.  Denied – means that the request has been denied by the absence administrator. |
| Main Two Weeks First Choice | An absence start date and absence end date for a member of staffs first choice for their main two week vacation. |
| Main Two Weeks Second Choice | An absence start date and absence end date for a member of staffs first choice for their main two week vacation. |
| Public Holidays | A list of dates for the year which are public holidays. All staff are absent on these days as the office is closed. |

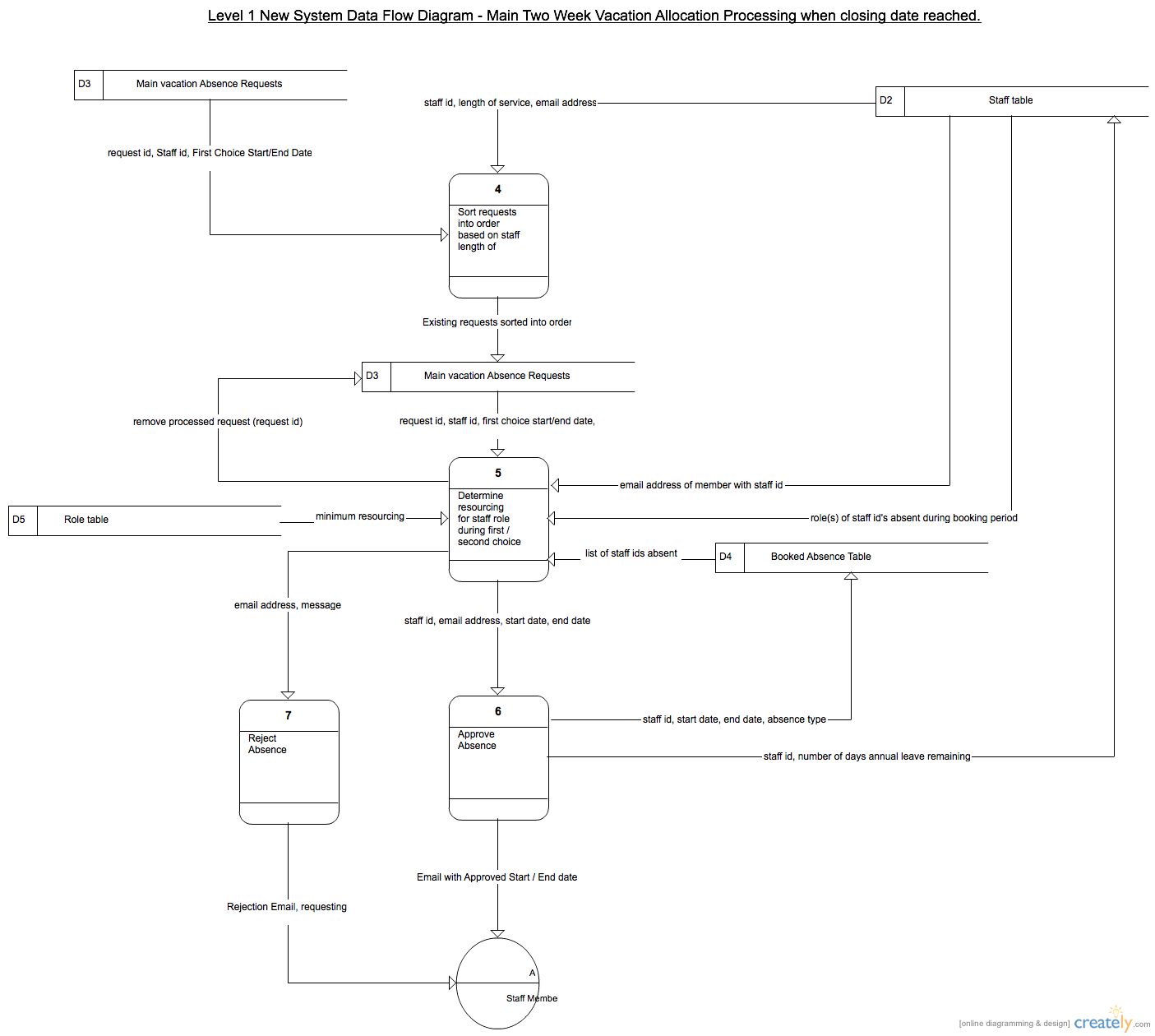
## Data flows

### Existing data flows



### Proposed data flows





## Objectives

The objective of my project is to design, implement, test and document an IT solution that provides automation of the current manual processes for absence administration within the company.

This system must:

* Maintain information on the various roles within the company, and the minimum resource level for each role that must be present on any given day.
* Maintain information on each employee within the company, including the length of service of the employee, and the role(s) which that employee is able to perform.
* Maintain information on the employees who are absent on each day of the year, together with the reason for absence.
* Provide a mechanism for staff to submit a first and second choice for their main annual vacation fortnight.
* Provide a mechanism to automatically allocate each member of staffs main two week vacation based on an algorithm involving the member of staffs length of service, the role that the member of staff performs, their first choice and second choice and the resourcing levels for that role during the requested period.
* Provide a mechanism for staff to submit ad-hoc absence requests throughout the year, and for the system to approve or reject these requests based on the type of request and the current resourcing levels during the requested period.
* Automatically calculate annual leave taken / remaining for each member of staff using an algorithm based on each absence booking, absence type and taking into account the fact that weekends and public holidays are non-working days.
* Provide a mechanism for the office manager or absence administrator to record unplanned absence (eg: if a member of staff phones in sick).
* Provide email generation to the office manager for any resource shortfalls due to unexpected absence.
* Provide the ability for a member of staff to view their current absence bookings for the year and annual leave days taken / remaining.
* Provide the ability for the office manager to view the absence bookings for all staff.
* Provide the ability for the absence administrator to view and amend absence bookings for all staff.
* Provide the ability for the absence administrator to define which days within a year are public holidays.

To implement these objectives I will utilise the following technologies:

* HTML5 and CSS to provide a web based user interface / presentation layer.
* Javascript and JQuery as the client side scripting language.
* Linux Server running Apache as the webserver.
* PHP as the server side scripting language.
* MySQL as the server side database technology.

The database will be based on a normalised logical model and consist of a number of tables.

The solution will include a login system, with different users having different roles (staff, administrator, office manager) and multiple web pages with functionality based on the role of the user.

The solution will include the design and implementation of a number of algorithms including:

* An algorithm to determine the optimal allocation of all staffs main vacation choices.
* An algorithm to calculate the number of days annual leave required for an absence request, taking into account that weekends and public holidays are non-working days.

## complexity

Based on the objectives and scope, I believe this is a complex project that will cover a large amount of the concepts in my A level computing course and provide me with the opportunity to develop my skills in a number of modern technologies used in the industry.

## potential solutions

**Using a Spreadsheet system**

Some elements of the current wall calendar could be represented in a spreadsheet.

|  |  |
| --- | --- |
| Pros | Cons |
| Commonly used tool, familiar to the absence administrator | Whilst aspects of the current processes could be automated, there would still be a large amount of manual data entry required by the absence administrator. |
|  | Does not easily allow for the bespoke algorithms required. |
|  | Does not enforce the different levels of user access / user visibility required. |

**Use a third party Online Solution**

There are a number of companies who provide an online absence management service, two of the main ones are:

<http://www.leaveplanner.com>

<http://www.whosoff.com>

|  |  |
| --- | --- |
| Pros | Cons |
| Available instantly | Specific algorithms required for main two week leave allocation not available ‘out of the box’ and current business processes may have to change to fit to the tool. |
| No overhead in developing or maintaining the system | Given the size of the company, any requests for changes to the tool are unlikely to be supported. |
| As a widely used tool, likely to be of a high quality with high availability. | Possible concerns over data security as all data is maintained by a third party, over the internet. |
|  | Monthly subscription fees. |

**Design and Implement a Bespoke Solution for the Company**

After analysis, continue with this project to design, implement, test and release a bespoke software solution to fully meet the needs of the company.

|  |  |
| --- | --- |
| Pros | Cons |
| Designed specifically to meet the needs of the company. | Requires bespoke development (this project). |
| Likely to provide the most automation of any solution. | Ongoing support costs for any future enhancements or fixes. |
| Can be tailored and extended as requested by the user. |  |

https://linuxacademy.com/cp/courses/lesson/course/2/lesson/7/module/1

Whilst the use of a spreadsheet may help to automate some of the current processes, it would still involve a lot of manual efforts on behalf of the holiday administrator, and therefore is not considered any further.

The use of commercial off the shelf solutions, examples of which were given in the previous section, are a viable option, but are unlikely to exactly match the specific requirements and algorithms that the company uses today. Therefore, company processes would need to be altered to fit to the tool. Also, these off the shelf solutions tend to be subscription based, meaning an on-going monthly cost to the company.

The proposed solution is therefore to proceed with my project to design and implement a custom solution for the company. To ensure the solution is easy to maintain, support and enhance in future.

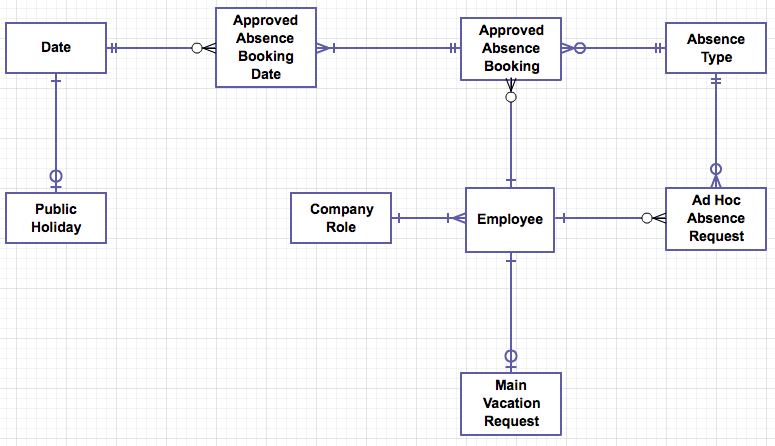
## use of formal methods

I plan to use formal analysis and design methods including Data Flow Diagrams to show the various flows of data and Entity relationship diagrams to define the structure of the database. I will prototype user interfaces and seek feedback from the end users.

I plan to use industry standard web based technologies for the project, including:

* HTML 5 / CSS for the presentation layer.
* Javascript / JQuery for the client side scripting.
* PHP for the server side scripting
* MySQL for the Database.

## entity relationship model



# Design

## Overall system design

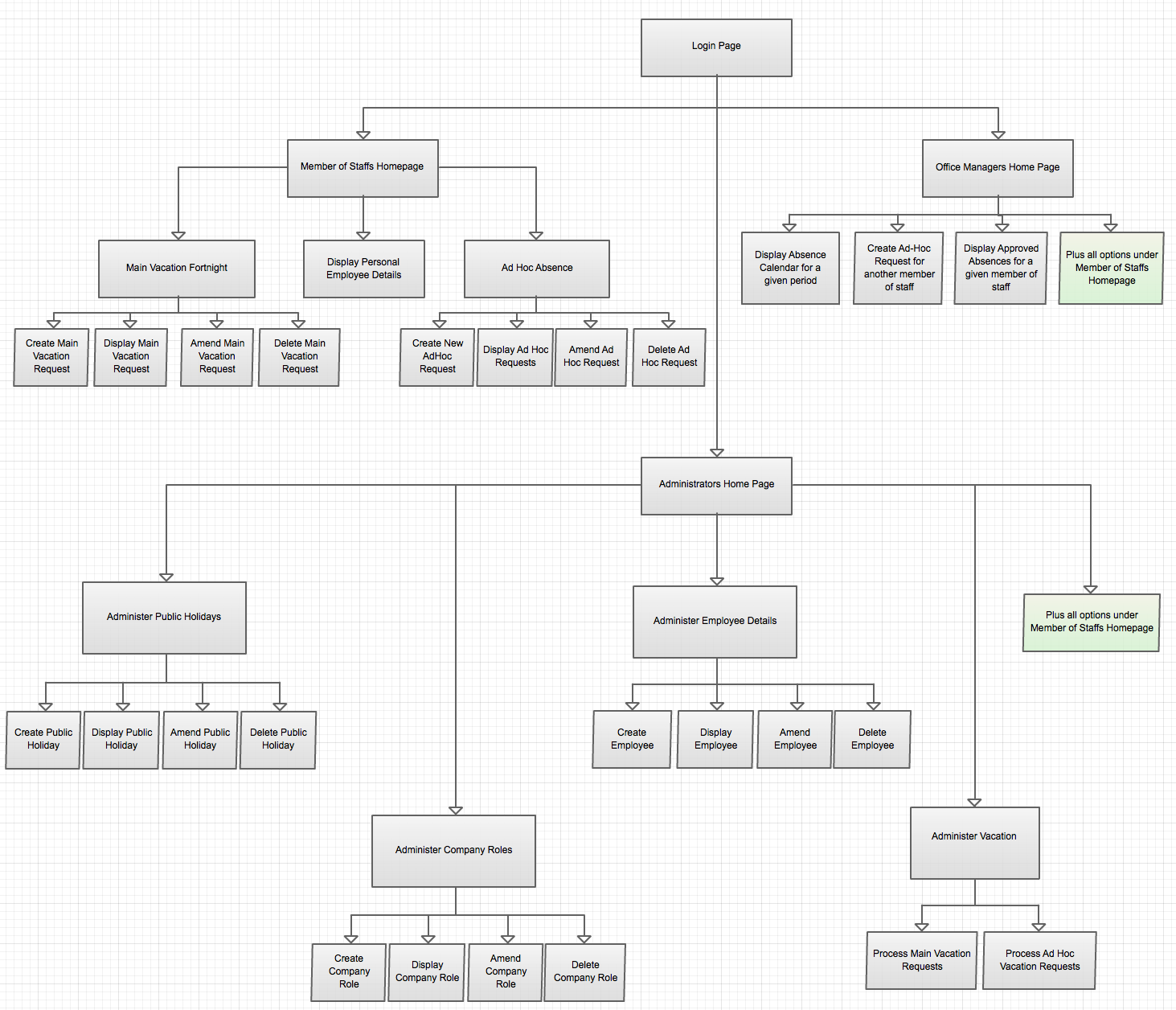
IOPS Chart

In the diagram below I have summarised the data of the system in terms of its input, output, processing and storage.

|  |  |
| --- | --- |
| **Input**  Employee Name Email Address Password  User Role System Role Date joined company Annual Leave entitlement  Main Vacation First Choice Start Date Main Vacation First Choice End Date Main Vacation Second Choice Start Date Main Vacation Second Choice End Date  Ad-Hoc Request Start Date Ad-Hoc Request End Date Type of Absence Request  Public Holiday Date Public Holiday Name  Role Name Minimum Staffing Level For Role  Absence Calendar Start Date Absence Calendar End Date | **Process**  Create New Employee Create Main Vacation Request Create Ad-Hoc Vacation Request Process Main Vacation Requests Process Ad-Hoc Vacation Requests Create Public Holiday Create Company Role  Display Employee Details Display Main Vacation Requests Display Ad-Hoc Vacation Requests Display Public Holiday Display Company Role  Amend Employee Amend Main Vacation Request Amend Ad-Hoc Vacation Request Amend Public Holiday Amend Company Role  Delete Employee Delete Main Vacation Request Delete Ad-Hoc Vacation Request Delete Public Holiday Delete Company Role  Display Absence Calendar for a given period Display Approved Absences for a given member of staff |
| **Storage**  Employee Table Main Vacation Request Table Ad Hoc Absence Request Table Public Holiday Table Company Role Table Employee Role Table Approved Absence Bookings Table Approved Absence Bookings Date Table Date Table | **Output**  Welcome email to employee with login details. Main Vacation Request Approval / Rejection email. Email to office manager to warn of staff resource issues. Ad Hoc Absence Request Approval / Rejection email. Employee Details display Main Vacation request display Ad Hoc Vacation request display Public Holidays display Company Roles display Absence Calendar display |

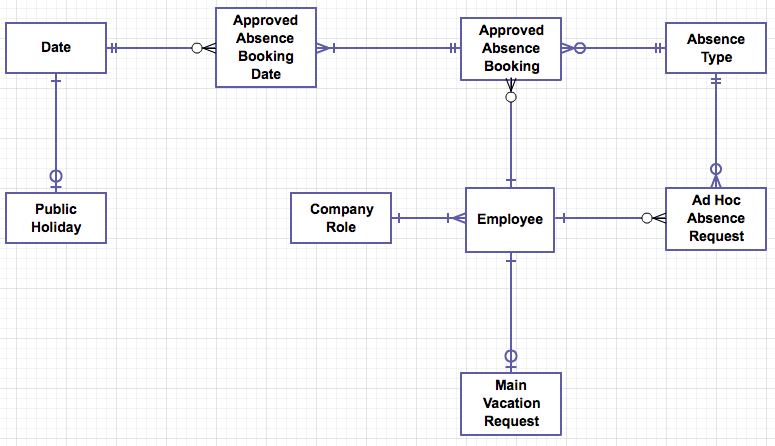
## Modular structure

I will structure my system around a series of web pages. The root page will be a login screen. Once the user has logged in, a homepage with the options relevant to that user will be displayed. There are three types of user: Member of Staff, Office Manager and Administrator.



## Database structure AND VALIDATION

The diagram below, taken from the analysis section, describes the relationship between each of the main tables in the database. In this section I will provide a description of the key fields in each table, together with the validation that will be performed on these fields on input.



### Public Holiday Table

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Field | Data type | length | validation check | validation description | valid data | invalid data |
| Public Holiday ID | INT | 4 | None. Automatically assigned by SQL upon creation. | Primary key. Automatically assigned by SQL on record creation. | N/A | N/A |
| Date ID | INT | 4 | Presence, ID Exists in Date Table | An ID must be supplied and the ID must exist as a key in the Date table. | Any value that matches an ID in the date table | (Blank) or any value which does not match an id in the date table. |
| Name of Public Holiday | String | 40 characters | Presence | String must not be empty. String must be less than 40 characters | Any string | (Blank) |

### Date Table

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Field | Data type | length | validation check | validation description | valid data | invalid data |
| Date ID | INT | 4 | None. Automatically assigned by SQL upon creation. | Primary key. Automatically assigned by SQL on record creation. | N/A | N/A |
| Date | DATE | 10 (DD-MM-YYYY) | Presence, Format | The date must be a valid calendar date. EG: Day number must be valid for the month given. Leap years need to be considered when validating February | Any Calendar date. | (Blank) or anything that is not a date. EG: 30-02-2014 |
| Public Holiday ID | INT | 4 | Optional,  ID exists in Public Holiday Table. | If public holiday ID is present, then it must match the ID key of an entry in the public holiday table | (Blank) or an ID matching an entry in the public holiday table. | Any ID which does not exist in the public holiday table |

### Approved Absence Booking Date Table

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Field | Data type | length | validation check | validation description | valid data | invalid data |
| Approved Absence Booking Date ID | INT | 4 | None. Automatically assigned by SQL upon creation. | Primary key. Automatically assigned by SQL on record creation. | N/A | N/A |
| Date ID | INT | 4 | Presence, ID Exists in Date table | An ID must be supplied and the ID must exist as a key in the Date table. | Any value that matches an ID in the date table | (Blank) or any value which does not match an id in the date table. |
| Approved Absence Booking ID | INT | 4 | Presence, ID Exists in Approved Absence Booking table | An ID must be supplied and the ID must exist as a key in the Approved Absence Booking table. | Any value that matches an ID in the Approved Absence Booking table | (Blank) or any value which does not match an id in the Approved Absence Booking table. |

### Approved Absence Booking Table

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Field | Data type | length | validation check | validation description | valid data | invalid data |
| Approved Absence Booking ID | INT | 4 | None. Automatically assigned by SQL upon creation. | Primary key. Automatically assigned by SQL on record creation. | N/A | N/A |
| Employee ID | INT | 4 | Presence, ID Exists in Employee table | An ID must be supplied and the ID must exist as a key in the Employee table. | Any value that matches an ID in the employee table | (Blank) or any value which does not match an id in the employee table. |
| Absence Start Date | DATE | 10 (DD-MM-YYYY) | Presence, Format | The date must be a valid calendar date. EG: Day number must be valid for the month given. Leap years need to be considered when validating February | Any Calendar date. | (Blank) or anything that is not a date. EG: 30-02-2014 |
| Absence End Date | DATE | 10 (DD-MM-YYYY) | Presence, Format Value | The date must be a valid calendar date. EG: Day number must be valid for the month given. Leap years need to be considered when validating February  The Date must be greater than or equal to the start date. | Any Calendar date which is greater than the start date | (Blank) or anything that is not a date. EG: 30-02-2014  Any value that is less than the start date. |
| Absence Type ID | INT | 4 | Presence, ID Exists in Absence Type table | An ID must be supplied and the ID must exist as a key in the Absence Type table. | Any value that matches an ID in the Absence Type table | (Blank) or any value which does not match an id in the Absence Type table |

### Absence Type Table

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Field | Data type | length | validation check | validation description | valid data | invalid data |
| Absence Type ID | INT | 4 | None. Automatically assigned by SQL upon creation. | Primary key. Automatically assigned by SQL on record creation. | N/A | N/A |
| Absence Type Name | String | 20 | Presence | A value must be provided, giving a name to this absence type: EG: “Training”, “Sickness” etc. | Any string | (Blank) |
| Uses Annual Leave | Boolean | 1 | TRUE or FALSE | Must either be set to TRUE or FALSE? | TRUE or FALSE | (Blank) or anything other than the valid values. |
| Can be Denied | Boolean | 1 | TRUE or FALSE | Must either be set to TRUE or FALSE? | TRUE or FALSE | (Blank) or anything other than the valid values. |

### Employee Table

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Field | Data type | length | validation check | validation description | valid data | invalid data |
| Employee ID | INT | 4 | None. Automatically assigned by SQL upon creation. | Primary key. Automatically assigned by SQL on record creation. | N/A | N/A |
| Employee Name | String | 50 | Is not empty | Check that a string has been given | Any string | NULL |
| Email Address | String | 50 | Is valid format for an email address | Must conform to the convention for an email address.  IE: name@address.ext | Any value that is a valid email address | Any other value. |
| Password | String | 20 | Minimum length  Letters and Numbers | Password must be at least 8 characters  Password must contain at least 1 number | Any string of 8 characters or more that has at least 1 number in the set of characters | Any string less than 8 characters in length or which does not contain at least 1 number. |
| Date Joined the Company | DATE | 10 (DD-MM-YYYY) | Presence, Format | The date must be a valid calendar date. EG: Day number must be valid for the month given. Leap years need to be considered when validating February | Any Calendar date. | (Blank) or anything that is not a date. EG: 30-02-2014 |
| Annual Leave Entitlement | INT | 1 | Maximum value check | Annual Leave Entitlement cannot exceed 35 days | Any value from 0 to 35 | Any value outside of this range |
| Main Vacation Request ID | INT | 4 | Optional, If present is a valid entry in the Main Vacation Request Table | If a value is given it must match an entry in the Main Vacation Request table | NULL or  Any ID value which matches an entry in the main vacation request table | Any value which does not match an entry in the main vacation request table |

### Ad Hoc Absence Request Table

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Field | Data type | length | validation check | validation description | valid data | invalid data |
| Ad Hoc Absence Request ID | INT | 4 | None. Automatically assigned by SQL upon creation. | Primary key. Automatically assigned by SQL on record creation. | N/A | N/A |
| Employee ID | INT | 4 | Is a valid Employee | ID given matches an Employee ID in the Employee Table | Any value that matches an Employee | Any other value. |
| Start Date | DATE | 10 (DD-MM-YYYY) | Presence, Format | The date must be a valid calendar date. EG: Day number must be valid for the month given. Leap years need to be considered when validating February | Any Calendar date. | (Blank) or anything that is not a date. EG: 30-02-2014 |
| End Date | DATE | 10 (DD-MM-YYYY) | Presence, Format Value | The date must be a valid calendar date. EG: Day number must be valid for the month given. Leap years need to be considered when validating February  The Date must be greater than or equal to the start date. | Any Calendar date which is greater than the start date | (Blank) or anything that is not a date. EG: 30-02-2014  Any value that is less than the start date. |
| Absence Type ID | INT | 4 | Is a valid Absence Type | ID given matches an Absence Type ID in the Absence Type Table | Any value that matches an Absence Type | Any other value. |

### Main Vacation Request Table

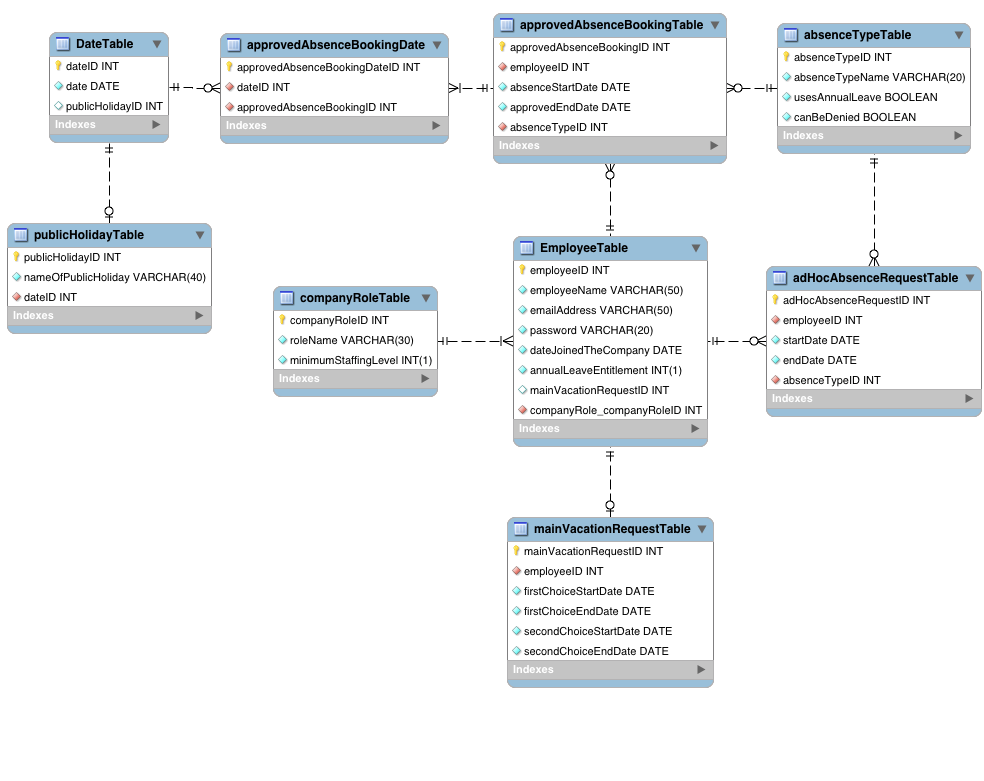
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Field | Data type | length | validation check | validation description | valid data | invalid data |
| Main Vacation Request ID | INT | 4 | None. Automatically assigned by SQL upon creation. | Primary key. Automatically assigned by SQL on record creation. | N/A | N/A |
| Employee ID | INT | 4 | Is a valid Employee | ID given matches an Employee ID in the Employee Table | Any value that matches an Employee | Any other value. |
| First Choice Start Date | DATE | 10 (DD-MM-YYYY) | Presence, Format | The date must be a valid calendar date. EG: Day number must be valid for the month given. Leap years need to be considered when validating February | Any Calendar date. | (Blank) or anything that is not a date. EG: 30-02-2014 |
| First Choice End Date | DATE | 10 (DD-MM-YYYY) | Presence, Format Value | The date must be a valid calendar date. EG: Day number must be valid for the month given. Leap years need to be considered when validating February  The Date must be greater than or equal to the first choice start date. | Any Calendar date which is greater than the first choice start date | (Blank) or anything that is not a date. EG: 30-02-2014  Any value that is less than the start date. |
| Second Choice Start Date | DATE | 10 (DD-MM-YYYY) | Presence, Format | The date must be a valid calendar date. EG: Day number must be valid for the month given. Leap years need to be considered when validating February | Any Calendar date. | (Blank) or anything that is not a date. EG: 30-02-2014 |
| Second Choice End Date | DATE | 10 (DD-MM-YYYY) | Presence, Format Value | The date must be a valid calendar date. EG: Day number must be valid for the month given. Leap years need to be considered when validating February  The Date must be greater than or equal to the second choice start date. | Any Calendar date which is greater than the second choice start date | (Blank) or anything that is not a date. EG: 30-02-2014  Any value that is less than the start date. |

### Company Role Table

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Field | Data type | length | validation check | validation description | valid data | invalid data |
| Company Role ID | INT | 4 | None. Automatically assigned by SQL upon creation. | Primary key. Automatically assigned by SQL on record creation. | N/A | N/A |
| Role Name | String | 30 | Presence | String, which is the name of a role within the company. Any string is valid. | Any string | (Blank) |
| MinimumStaffingLevel | INT | 1 | Minimum number of staff for this role who need to be in the office on any given day. | Positive integer. | Any positive integer | Blank or a negative integer |

## file organisation and processing

## database design and entity relationship model



* + 1. **Public Holiday Table – Date Table**

The relationship between the Public holiday and date tables show that each public holiday must be tied to a date (E.G December 25th is a holiday). However, a date can have either have a public holiday tied to it or no public holiday. Therefore we can state that a date is tied to 0 or 1 public holiday.

* + 1. **Date Table - Approved Absence Booking Date Table**

The relationship between the Date and Approved Absence Booking Date shows that each Approved Absence Booking Date can only be tied to one date. This is done so that no single employee is able to accidentally book the same date off twice. However, one date can have 0 or many different Approved Absence Booking Dates. This is done to allow more than one employee to have the Approved Date off. This is possible if there are enough staff to cover minimum levels in one role or if the employees are both used in different company roles.

* + 1. **Approved Absence Booking Date Table – Approved Absence Booking Table**

The relationship between the Approved Absence Booking Date and the Approved Booking Date tables show that each Approved Absence Booking Date is tied to a single Approved Absence Booking. However, An Approved Absence Booking can have 1 or multiple Approved Absence Booking Dates. This will allow employees to book multiple days off, which is required for allowing an employee’s main two weeks, but is also required for extended periods of leave in other situations, such as sickness, compassionate leave etc.

* + 1. **Approved Absence Booking Table – Absence Type Table**

The relationship between the Approved Absence Booking and the Absence Type tables show   
that an absence type can apply to 1 or multiple Approved Absence Bookings, but an Approved Absence Booking can only have 1 Absence Type. As an employee will only book off a date for one reason, the Approved Absence Booking table should only be able to store one Absence Type.  
However, many absence types may apply to multiple Approved Absence Bookings at any one time.

* + 1. **Absence Type Table – Ad Hoc Absence Request Table**

The relationship between the Absence Type and the Ad Hoc Absence Request tables show that each Ad Hoc Absence Request is tied to a single Absence Type. However, An Ad Hoc Absence Booking can have 0 or multiple Ad Hoc Absence Requests. The reasoning for this are very similar to that of the Approved Absence Booking – Absence Type

* + 1. **Employee Table – Approved Absence Booking Table**

The relationship between the Employee and the Approved Absence Booking tables show that each Employee can be tied to 0 or multiple Approved Absence Bookings, but every Approved Absence Booking is tied to one employee. This allows the employees taking the absences can be easily identified so as to allow the number of days leave available to drop (Except in special circumstances.

* + 1. **Employee Table – Ad Hoc Absence Request Table**

The relationship between the Employee and the Ad Hoc Absence Request tables show that each Ad Hoc Absence Request Booking Date is tied to a single Employee. Each Ad Hoc Absence Request can have 0 or many Employees.

* + 1. **Employee Table – Main Vacation Request Table**

The relationship between the Employee and the Main Vacation tables show that each Main Vacation Request is tied to a single Employee and each employee can only have one main vacation request.

* + 1. **Employee Table – Company Role Table**

The relationship between the Employee and the Company Role tables show that each Employee has a single role within the company, but that a single role in the company may be performed by one or more employees.

## storage media and format

I will store the data for the application in a MySQL database, using a number of tables to logically represent the main aspects of the system, and relationships between these.

## algorithms DESIGN

### Annual Leave required for a booking Algorithm

In order to ensure that the user does not book more vacation than they are entitled to, for any given absence booking, we need to know the number of days annual leave that will be required for that booking.

No annual leave is required when the type of absence requested does not use annual leave (for example sick leave). We can determine whether a type of absence requires annual leave from the usesAnnualLeave flag in the AbsenceTypeTable entry.

If the type of absence request does use annual leave, then each day from start date to end date should be examined. Saturday and Sundays do not consume annual leave and any public holidays in the booking period will not consume annual leave, any other days will.

The pseduocode below is for a function which will take the start date, end date and absence type as input parameters and return an integer, which represents the number of annual leave days required for this booking.

|  |
| --- |
| Pseudo Code |
| FUNCTION CalculateAnnualLeaveRequired(startDate, endDate,absenceType, publicHolidayDateArray)  annualLeaveRequired ← 0  IF absenceType.usesAnnualLeave = TRUE THEN  tempDate ← startDate    WHILE tempDate <= endDate  IF DayOfWeek(tempDate) <> Saturday AND DayOfWeek(TempDate) <> Sunday THEN  isPublicHoliday ← FALSE  FOR arrayIndex← 0 TO publicHolidayDataArray.length  IF publicHolidayDateArray[arrayIndex] = tempDate THEN  isPublicHoliday ← TRUE  ENDIF  ENDFOR   IF IsPublicHoliday = FALSE THEN  AnnualLeaveRequired ← AnnualLeaveRequired + 1  ENDIF  ENDIF  tempDate ← nextDay(tempDate)  ENDWHILE  ELSE  AnnualLeaveRequired ← 0  ENDIF  RETURN AnnualLeaveRequired  ENDFUNCTION |

Note: DayOfWeek is available in PHP through the $DayOfWeek = date(“D”,TempDate);

|  |  |  |
| --- | --- | --- |
| Annual Leave Remaining Algorithm In order to ensure that the user does not book more vacation than they are entitled to, we need to know the number of remaining days annual leave that a member of staff has.  To do this we need to start with the count set to the employee’s annual leave entitlement, obtain a list of all of the approved absence bookings for the employee and then for each booking, use the earlier CalculateAnnualLeaveRequired function to determine the number of days needed for the booking, subtracting this from the count.  Once we have done this for each booking, the count will represent the number of remaining days annual leave available for the employee.   |  | | --- | | Pseudo Code | | FUNCTION CalculateRemainingAnnualLeave (paramEmployeeID)  annualLeaveRemaining ← employeeID.annualLeaveEntitlement  arrayOfAbsenceBookings ← Select \* from ApprovedAbsenceBookingTable where employeeId = paramEmployeeID  publicHoldidayDateArray ← select date from DateTable where publicHolidayId != NULL  FOR arrayIndex← 0 TO arrayOfAbsenceBooking.length  startDate ← arrayOfAbsenceBookings[arrayIndex].startDate  endDate ← arrayOfAbsenceBookings[arrayIndex].endDate  absenceType ← arrayOfAbsenceBookings[arrayIndex].absenceType  leaveRequired ← calculateAnnualLeaveRequired(startDate,endDate,absenceType,publicHolidayDateArray)  annualLeaveRemaining ← annualLeaveRemaining – leaveRequired  ENDFOR  RETURN annualLeaveRemaining  ENDFUNCTION | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Has Sufficient Annual Leave Algorithm When processing an absence request, the first check that will need to be made is whether the member of staff has sufficient annual leave available to cover the absence period requested.  The algorithm to do this can make use of the earlier algorithms defined.  Given an employee ID, Start Date, End Date and Absence Type, we will:   1. Calculate the annual leave remaining for the employee, using the Annual Leave Remaining algorithm described earlier. 2. Calculate the amount of annual leave required for the booking, using the Calculate Annual Leave Required algorithm described earlier. 3. Check to ensure that the leave required is less than or equal to the employees remaining annual leave. If it is, the function returns TRUE, otherwise it returns FALSE.  |  | | --- | | Pseudo Code | | FUNCTION HasSufficentAnnualLeave (paramEmployeeID, startDate,endDate,absenceType)  hasSufficentLeave ← FALSE  employeesAvailableLeave ← CalculateRemainingAnnualLeave(paramEmployeeID)  amountOfLeaveNeeded ← CalculateAnnualLeaveRequired(startDate,endDate,absenceType)    IF amountOfLeaveNeeded <= employeesAvailableLeave THEN  hasSufficentLeave ← TRUE  ENDIF  RETURN hasSufficentLeave ENDFUNCTION |  Are Sufficient Staff in Role to Grant Request Algorithm As part of processing any absence request, a check needs to be made to determine whether the request would cause the number of staff in a role to fall below the minimum staffing level.  As input to this algorithm, we need the ID of the employee requesting the absence and the start and end dates of the absence period.  Firstly we need to get the ID of the company role that the employee performs, and the minimum staffing level for that role. Also, we need to count up the total number of staff in the company who perform that role.  We then need to iterate through each date in the period from start date to end date.  For each date we need to:   1. Set the temp count of employees in role to total number of staff in company who perform the role. 2. Get the ID of the date from the date table. 3. Using the date ID, get a list of all approved absence booking ids for that date from the ApprovedAbsenceBookingDate table. 4. For each approved absence booking, get the employee ID who has booked that date. 5. Using the employee ID, get their role. 6. If the role is the same as the role of the employee requesting the absence then decrease the temp count. 7. If the temp count becomes less than or equal to the minimum staffing level for the role, then there are insufficient staff in the role to grant the request.  |  | | --- | | Pseudo Code | | FUNCTION SufficentStaffInRoleToGrantRequest (paramEmployeeID, startDate,endDate)  sufficentStaffInRole ← TRUE  employeeRole ← select companyRoleID from EmployeeTable where employeeID = paramEmployeeID   minimumStaffingLevel ← select minimumStaffingLevel from CompanyRoleTable where companyRoleID = employeeRole  numOfEmployeesInRole ← select count(employeeID) from EmployeeTable where companyRoleID = employeeRole ←  tempDate ← startDate  underMinimumStaffing ← FALSE  WHILE tempDate <= endDate AND underMinimumStaffing = FALSE  tempStaffingLevel ← numOfEmployeesInRole  dateID ← select dateID from DateTable where date = tempDate  bookingsForDate ← select approvedAbsenceBookingID from approvedAbsenceBookingDateTable   where approvedAbsenceBookingDate.dateID = dateID  FOR index ← 0 to bookingsForDate.length  bookersID ← select employeeID from approvedAbsenceBookingTabke where approvedAbsenceBookingID = bookingIDArray[index]  bookersRoleID ← select companyRole from EmployeeTable where Employee.employeeID = bookersID   IF employeeRole = bookersRoleID THEN   tempStaffingLevel ← tempStaffingLevel -1   ENDIF   ENDFOR   IF tempStaffingLevel <= minimumStaffingLevel THEN  underMinimumStaffing ← TRUE  sufficentStaffInRole ← FALSE  ENDIF  tempDate ← nextDay(tempDate)  ENDWHILE  RETURN sufficentStaffInRole  ENDFUNCTION | |
|  |

### Process Absence Request Algorithm

An absence request will have an employee ID, a start date, and end date and an absence type. To process the request, first we must check to ensure that the employee has sufficient annual leave remaining to cover the period requested. To do this we can make use of the hasSufficentAnnualLeave algorithm defined earlier.

If there is insufficient leave, a message will be given to the employee stating that the request is denied due to insufficient leave remaining.

If there is sufficient leave, then we need to check to see if there are sufficient staff of the same role as the employee available during the period requested. We can make use of the sufficentStaffInRoleToGrantRequest algorithm defined earlier to perform this check.

If there are sufficient staff in role, then the request is granted and we can remove the entry from the absence request table, and create entries in the approved absence booking table and approved absence booking date table.

If there is not sufficient staff in role, then we need to check whether the type of absence request can be denied. Certain absence requests, such as sick leave, can not be denied. If the request can be denied it is and a message returned to the employee informing them. If the request can not be denied, then it is approved, but also an email will be sent to the office manager alerting them to a staff shortage.

|  |
| --- |
| Pseudo Code |
| FUNCTION ProcessAbsenceRequest (paramEmployeeID, startDate,endDate,absenceType)  bookingApproved ← TRUE  IF HasSufficentAnnualLeave (paramEmployeeID, startDate, endDate, absenceType) = FALSE THEN  sendEmailToEmployee( “Request Denied. Insufficient annual leave remaining.”)  bookingApproved ← FALSE  ELSE  IF SufficentStaffInRoleToGrantRequest(paramEmployeeID, startDate, endDate, absenceType) THEN  Create new entry in Approved Absence Booking Date Table  Create new entry in Approved Absence Booking Table  sendEmailToEmployee( “Request Approved.”)  bookingApproved ← TRUE  ELSE  IF absenceType.canBeDenied THEN   bookingApproved ← FALSE  sendEmailToEmployee( “Request Denied. Request would leave role below minimum staffing level.”)  Remove entry from AdHoc Request Table    ELSE  sendEmailToManager(“Warning: below minimum staffing levels.”)  sendEmailToEmployee( “Request Approved.”)  Create new entry in Approved Absence Booking Date Table  Create new entry in Approved Absence Booking Table  bookingApproved ← TRUE  ENDIF   ENDIF  ENDIF  RETURN bookingApproved  ENDFUNCTION |

### Main Vacation Request Allocation Algorithm

When all main vacation requests have been received, the absence administrator will ask the system to process the main vacation requests.

To do this, the system must sort the vacation request into order, based on each member of staff’s length of service (longest to shortest).

The system will then iterate through this sorted set of requests.

For each request the system will check to see if the first choice period requested is allowed. It can do this using the ProcessAbsenceRequest algorithm defined earlier.

If the first choice is denied, then the ProcessAbsenceRequest algorithm will be called again, using the second choice dates.

If neither the first or second choice can be granted, the system will inform the member of staff concerned by email, asking them to resubmit new dates. The systems will also inform the absence administrator.

|  |
| --- |
| Pseudo Code |
| FUNCTION ProcessMainVacationRequests ()  mainVacationRequestsArray ← SELECT mainVacationRequest.\*,Employee.dateJoinedTheCompany FROM mainVacationRequest   INNER JOIN Employee ON mainVacationRequest.employeeID=Employee.employeeID   ORDER BY Employee.dateJoinedTheCompany ASC  numOfRequests ← mainVacationRequestsArray.length  FOR index←0 TO numOfRequests  IF ProcessAbsenceRequest(mainVacationRequestArray[index].employeeID,  mainVacationRequestArray[index].firstChoiceStartDate,  mainVacationRequestArray[index].firstChoiceEndDate,  Absence Type for Annual Leave) = FALSE THEN    IF ProcessAbsenceRequest(mainVacationRequestArray[index].employeeID,  mainVacationRequestArray[index].secondChoiceStartDate,  mainVacationRequestArray[index].secondChoiceEndDate,  Absence Type for Annual Leave) = FALSE THEN  sendEmailToEmployee(“Unable to grant either first or second choice for main vacation. Please submit new request.”)  sendEmailToAdministrator(“Main vacation request processing halted. Awaiting new dates from employee.”)  ENDIF  ENDIF    Delete Entry from mainVacationRequest Table.  NEXT  ENDFUNCTION |

### Ad Hoc Request Processing Algorithm

Processing Ad Hoc requests is very similar to Main Vacation Requests.

To do this, the system must select all entries from the AdHocAbsenceRequest table.

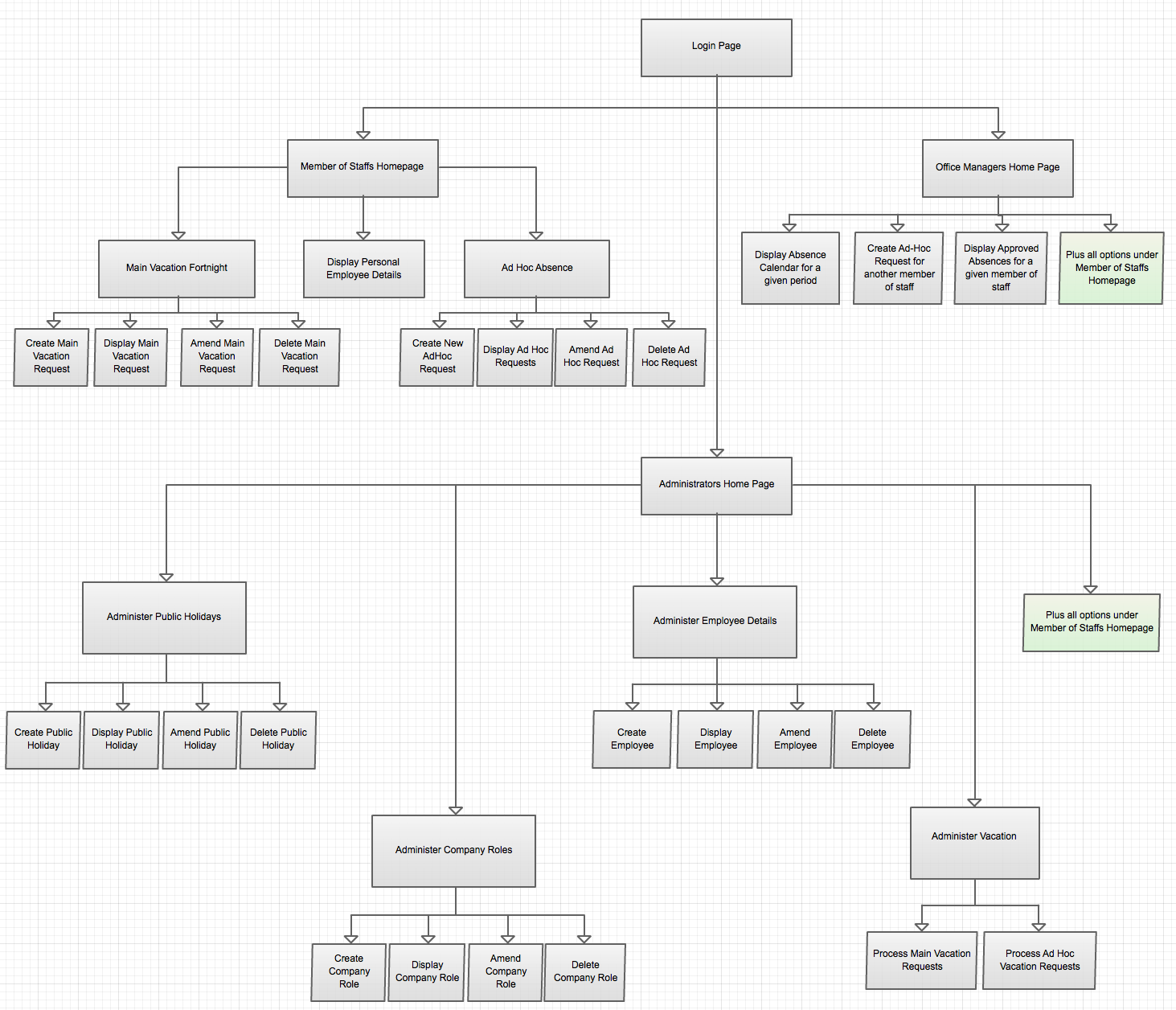
The system will then iterate each of these requests.

For each request the system will check to see if period requested is allowed. It can do this using the ProcessAbsenceRequest algorithm defined earlier.

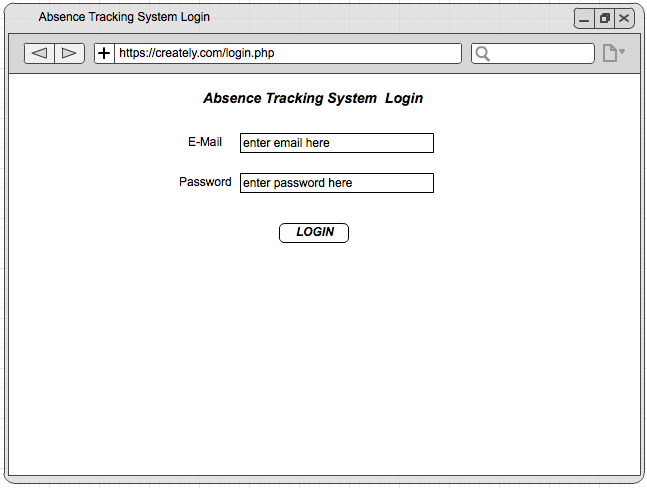
Once a request has been processed, the entry can be deleted from the AdHocAbsenceRequest table.

|  |
| --- |
| Pseudo Code |
| FUNCTION ProcessAdHocAbsenceRequests ()  AdHocRequestsArray ← SELECT \* FROM AdHocAbsenceRequest  numOfRequests ← AdHocRequestsArray.length  FOR index←0 TO numOfRequests  ProcessAbsenceRequest(AdHocAbsenceRequestsArray [index].employeeID,  AdHocAbsenceRequestsArray [index].startDate,  AdHocAbsenceRequestsArray [index].endDate,  AdHocAbsenceRequestsArray [index].absenceTypeID)  Delete Entry from mainVacationRequest Table.  NEXT  ENDFUNCTION |

## user interface



### Login Page

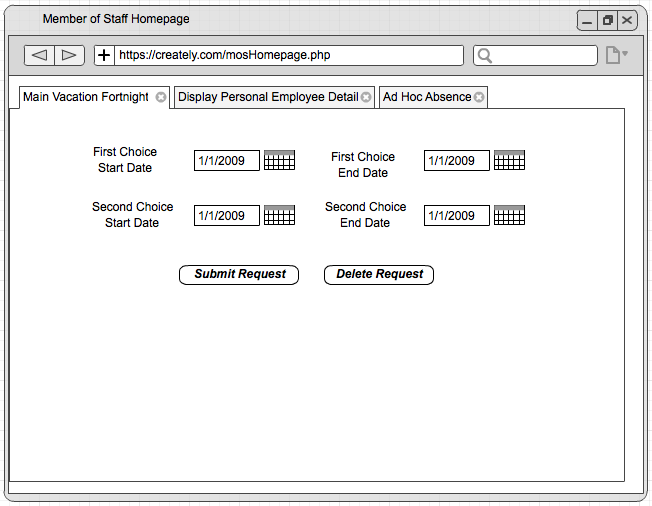


Loading this page will invoke the function LoginPageInit.

Clicking the login button will invoke the function LoginPageSubmit.

### Member of Staffs Homepage

#### Main Vacation Request

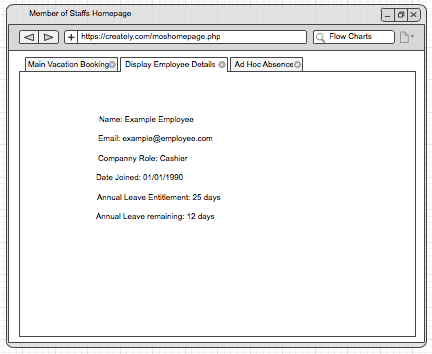


Loading this page will invoke the function MainVacationPageInit.

Clicking the submit request button will call the function MainVacationPageSubmit.

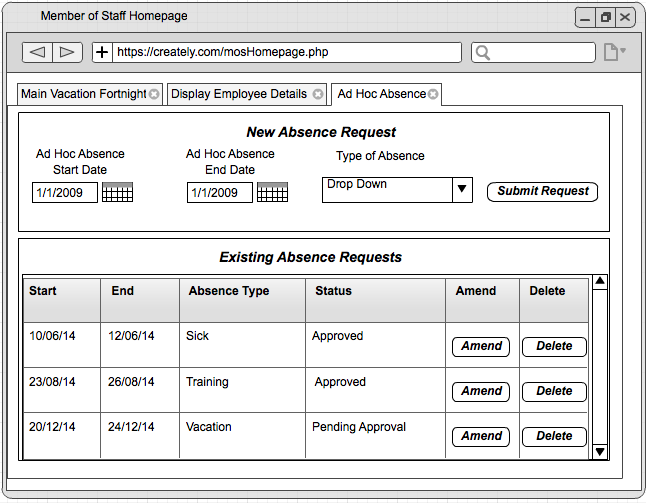
Clicking the delete request button will call the function MainVacationPageDelete.

#### Display Personal Employee Details



Loading this page will invoke the function EmployeeDetailsInit.

#### Ad Hoc Absence



Loading this page will invoke the function AdHocAbsenceInit.

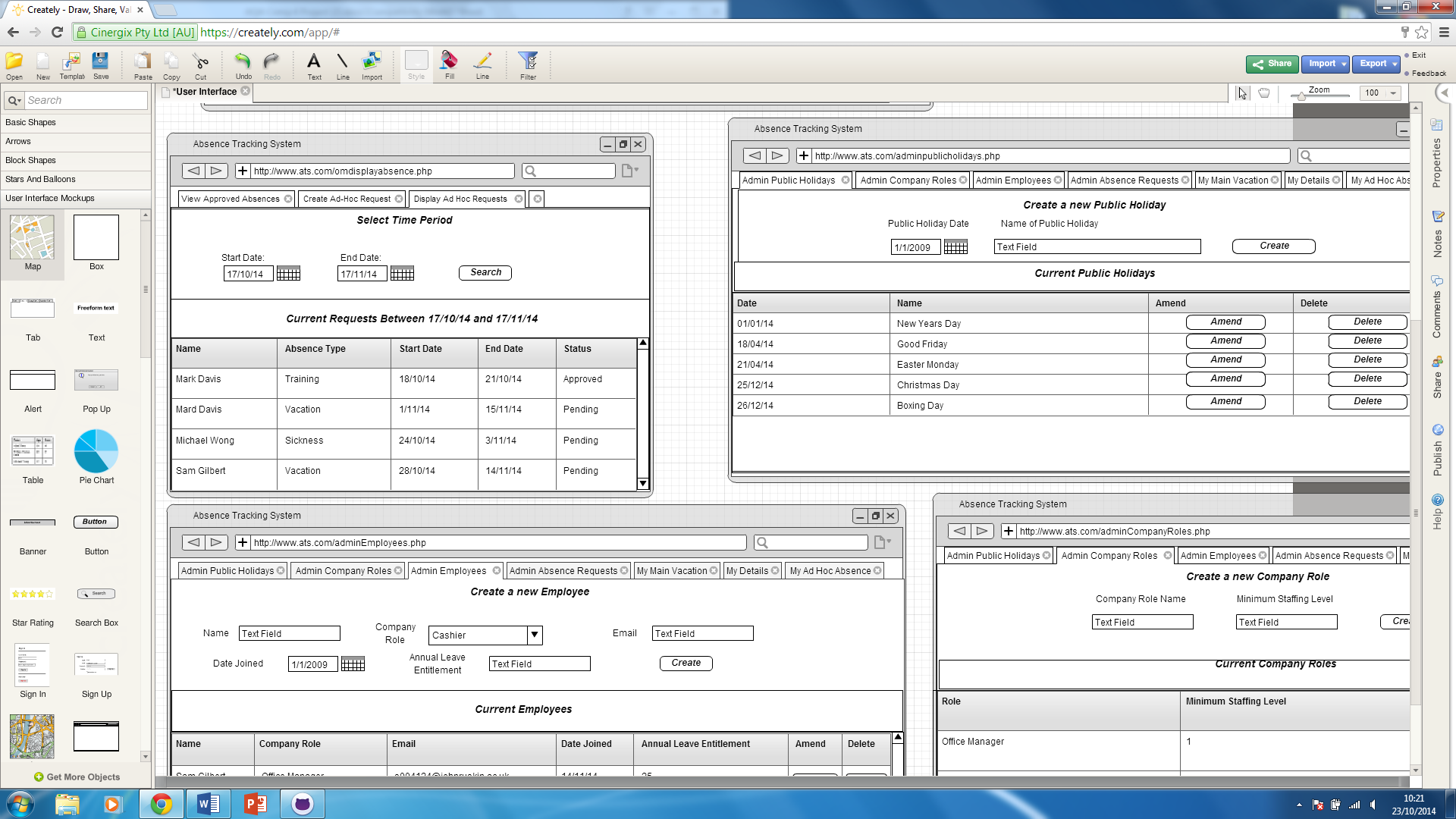
Clicking the submit request button will call the function AdHocAbsenceSubmit.

Clicking the amend request button will call the function AdHocAbsenceAmend.

Clicking the delete request button will call the function AdHocAbsenceDelete.

### Office Managers Homepage

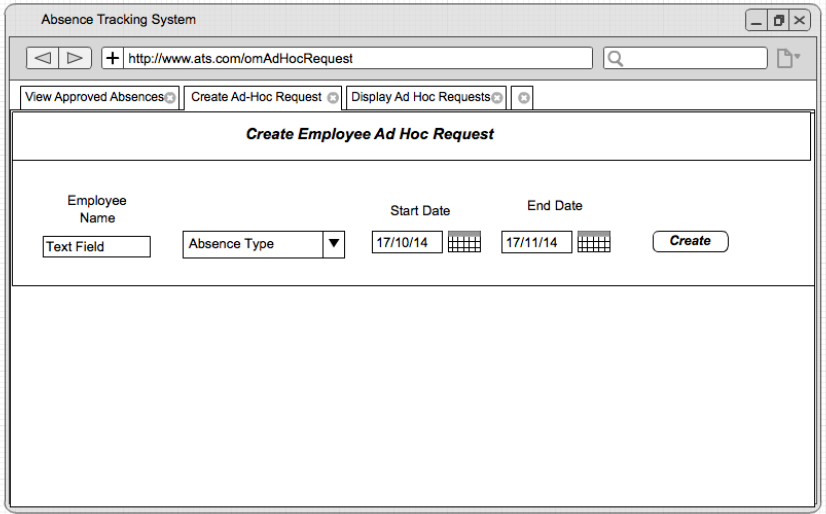
#### View Approved Absence Request Table



Loading this page will invoke the function OMApprovedAbsenceInit.

Clicking the search button will call the function OMApprovedAbsenceSearch.

#### Ad Hoc Absence for Other Members of Staff



Loading this page will invoke the function OMCreateAbsenceInit.

Clicking the search button will call the function OMCreateAbsenceSubmit.

#### Main Vacation Fortnight

See 3.8.2.1

#### Display Personal Employee Details

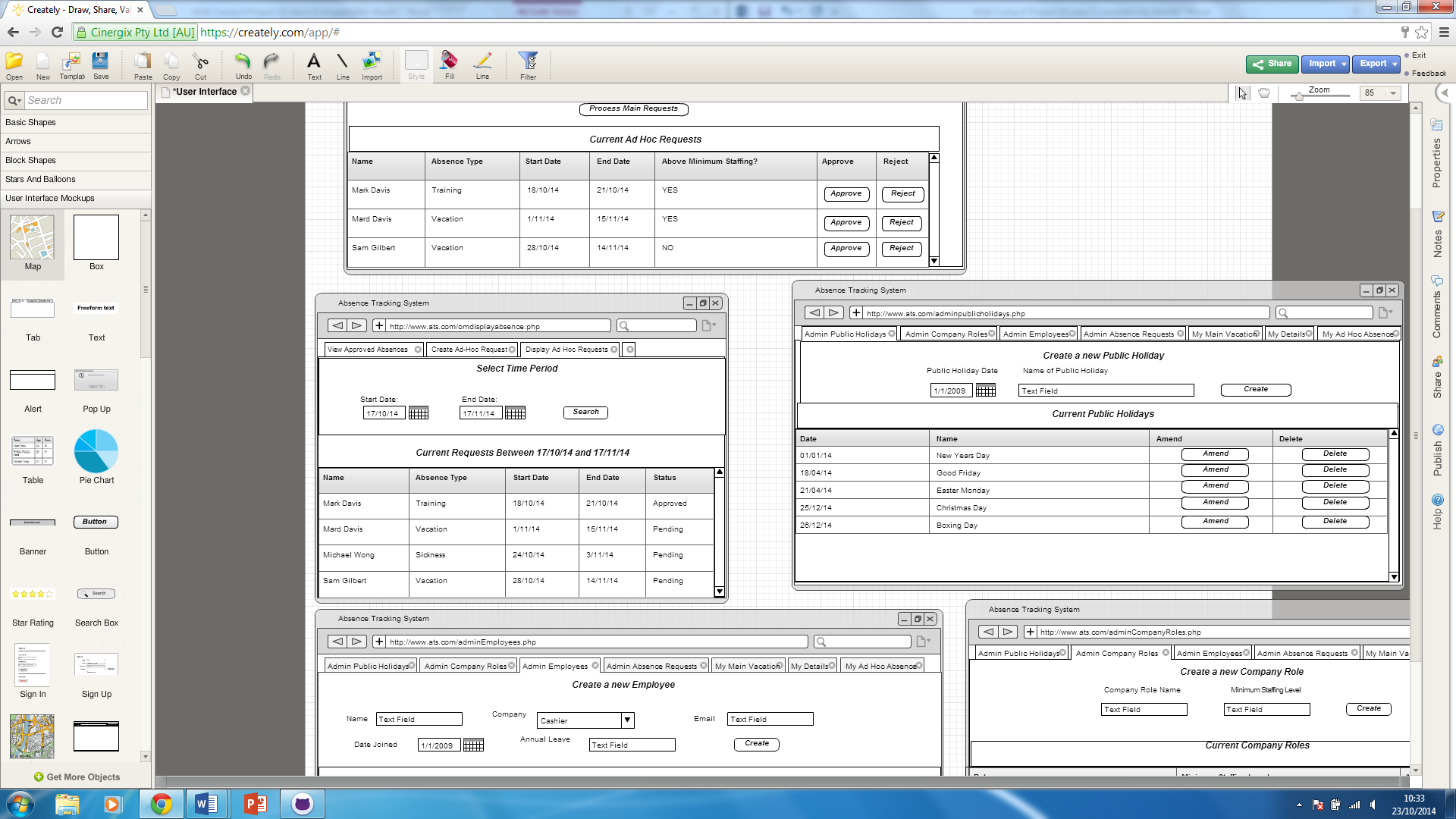
See 3.8.2.2

#### Ad Hoc Absence

See 3.8.2.3

### Administrators Homepage

#### Administer Public Holidays



Loading this page will invoke the function AdminPublicHolidayInit.

Clicking the Create button will call the function AdminPublicHolidayCreate.

Clicking the Amend button will call the function AdminPublicHolidayAmend.

Clicking the Delete button will call the function AdminPublicHolidayDelete.

#### Administer Company Roles



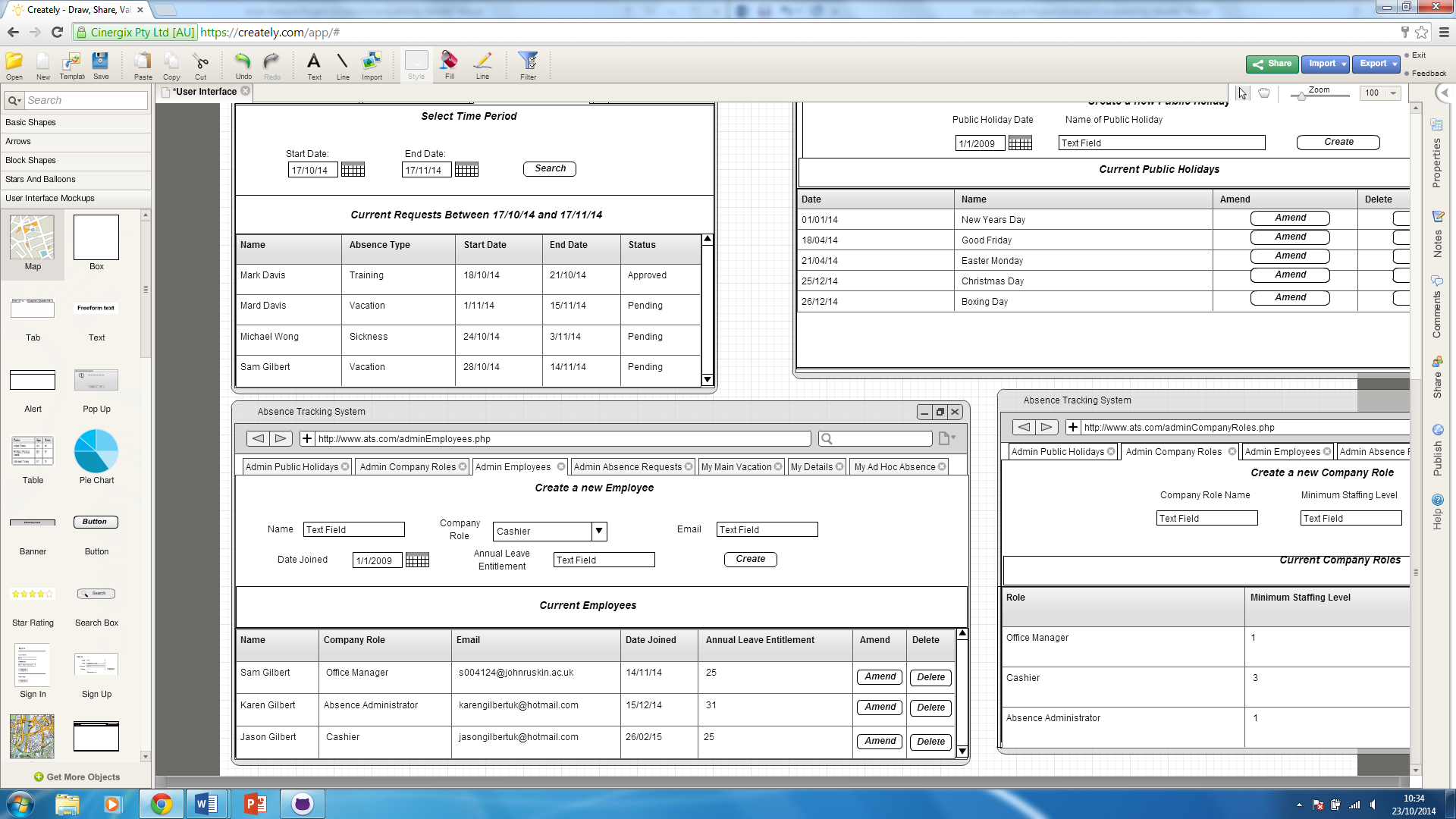
Loading this page will invoke the function AdminRoleInit.

Clicking the Create button will call the function AdminRoleCreate.

Clicking the Amend button will call the function AdminRoleAmend.

Clicking the Delete button will call the function AdminRoleDelete.

#### Administer Employee Details



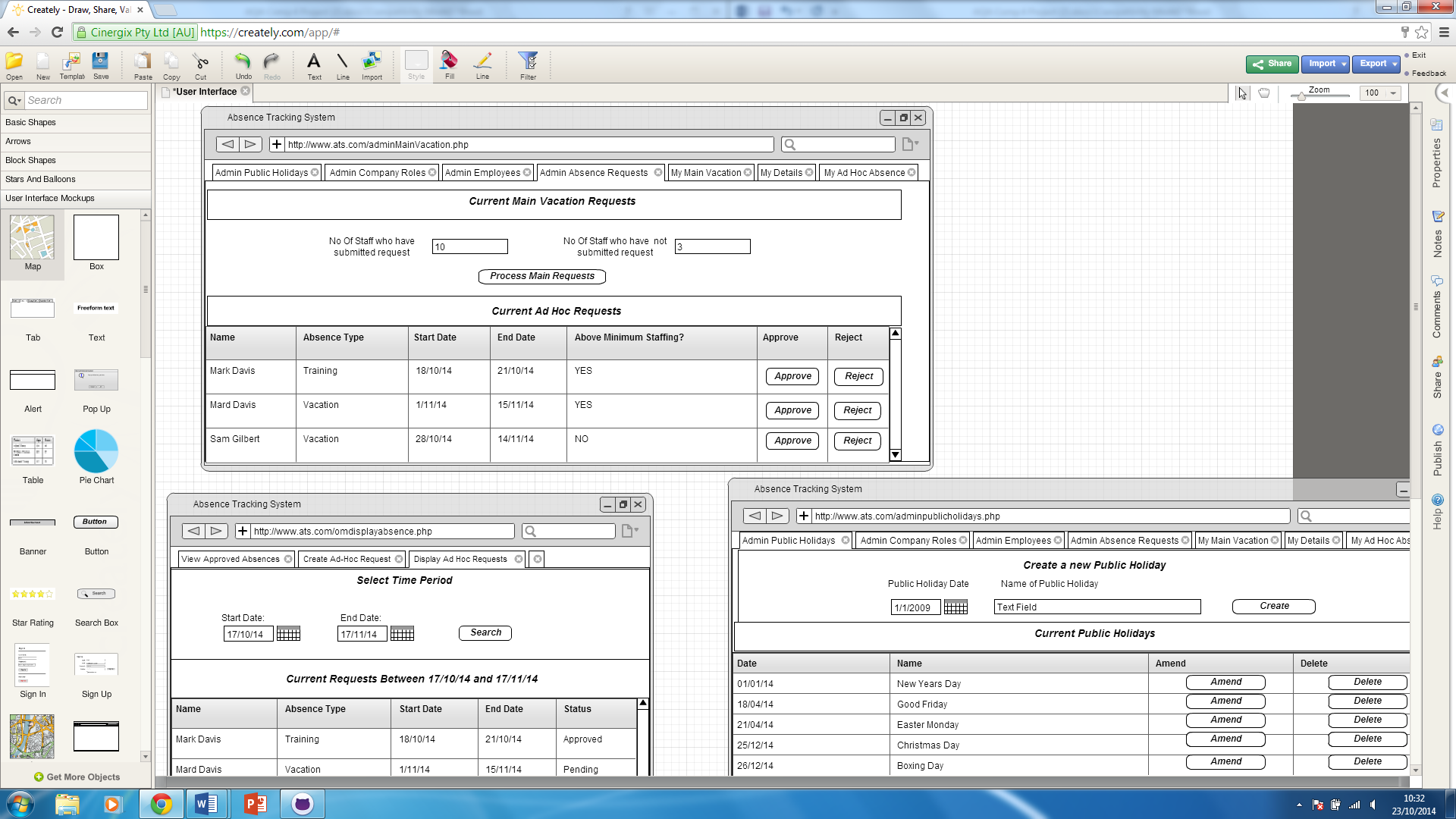
Loading this page will invoke the function AdminEmployeeInit.

Clicking the Create button will call the function AdminEmployeeCreate.

Clicking the Amend button will call the function AdminEmployeeAmend.

Clicking the Delete button will call the function AdminEmployeeDelete.

#### Administer Vacation



Loading this page will invoke the function AdminMainVacationInit.

Clicking the Process Main Requests button will call the function AdminMainVacationProcess.

Clicking the Approved button will call the function AdminAdHocApprove.

Clicking the Reject button will call the function AdminAdHocReject.

#### Main Vacation Fortnight

See 3.8.2.1

#### Display Personal Details

See 3.8.2.2

#### Ad Hoc Absence Request

See 3.8.2.3

## Security and integrity of data

### Data Security

The Main portion of the systems data security comes from the login and password system.

In my implementation I plan to use the concept of session id to identify the user once they have logged into the system. Without a valid session id, a user will not be able to interact with the application or access any of the data.

When a user first visits the website, or if their session has timed out, then they will not have a session id and therefore will be directed to the login page.

The login page will ask for their email and password.

Email and password are fields within the employee table.

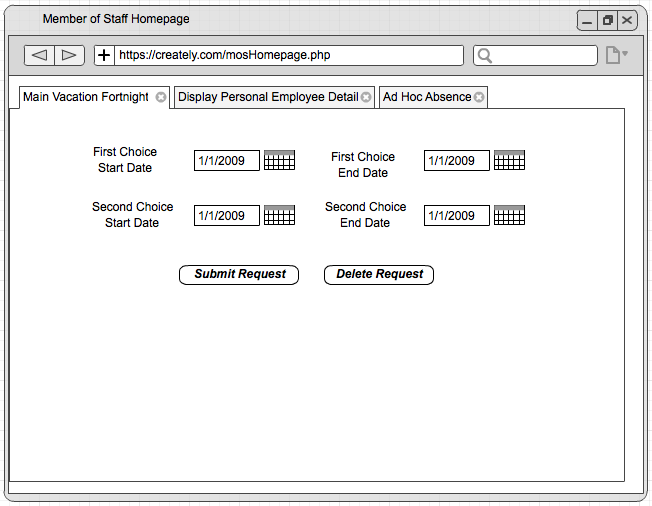
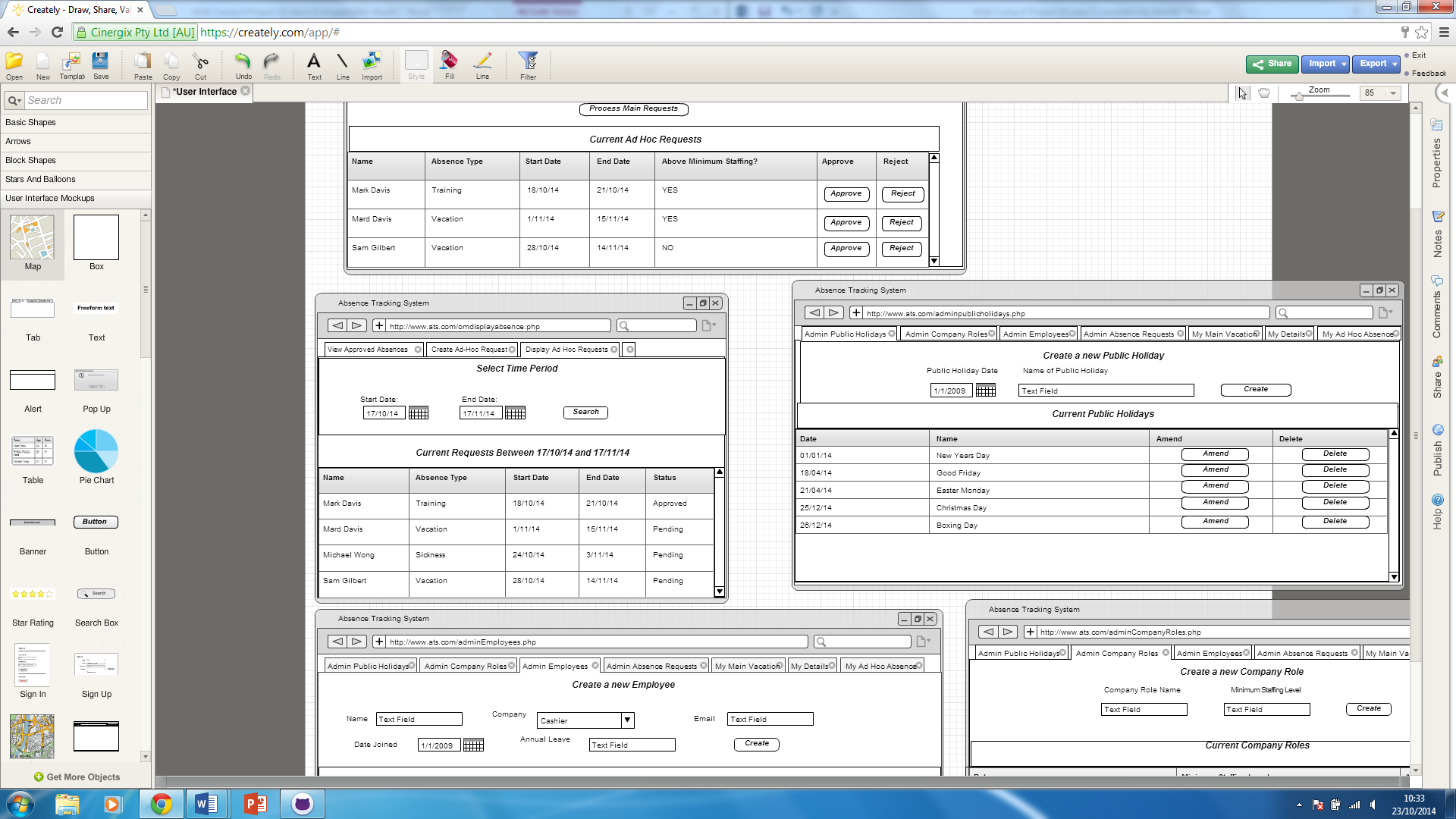
When a user sets their password I will employ logic to ensure that the password meets a minimum level of security standards. Namely:

1. The password must contain at least one upper case letter.
2. The password must contain at least one numeric character.
3. The password must be a minimum of 8 characters in length.

In the employee table I will not store the password in plain text, since this would allow anyone with direct access to the database to obtain users passwords.

Instead, when storing passwords in the database, I will use MD5 encryption to encode the password based on a hash derived from the email address.

When a valid username and password is input to the login page, the PHP on the page will fetch the users role from the SQL Database. Using this data, the website will either display a generic Member of Staff Homepage, an Office Manager Homepage or an Admin Homepage. An example of which can be seen below.



### Data Integrity

All data will be held in a MySQL database. In the implementation of my project I will create a set of functions that encapsulate all access to the database. To protect the integrity of the data, all changes to data in the database will be subject to strict validation rules, both at the user interface level, but also within the PHP scripts to process these requests.

### System Security

I plan to create an automatic scheduled nightly backup of the database. In the user guide I will include instructions for the administrator to follow in order to restore the database from a previous backup. This procedure would only be required in the event of a database corruption.

## test strategy

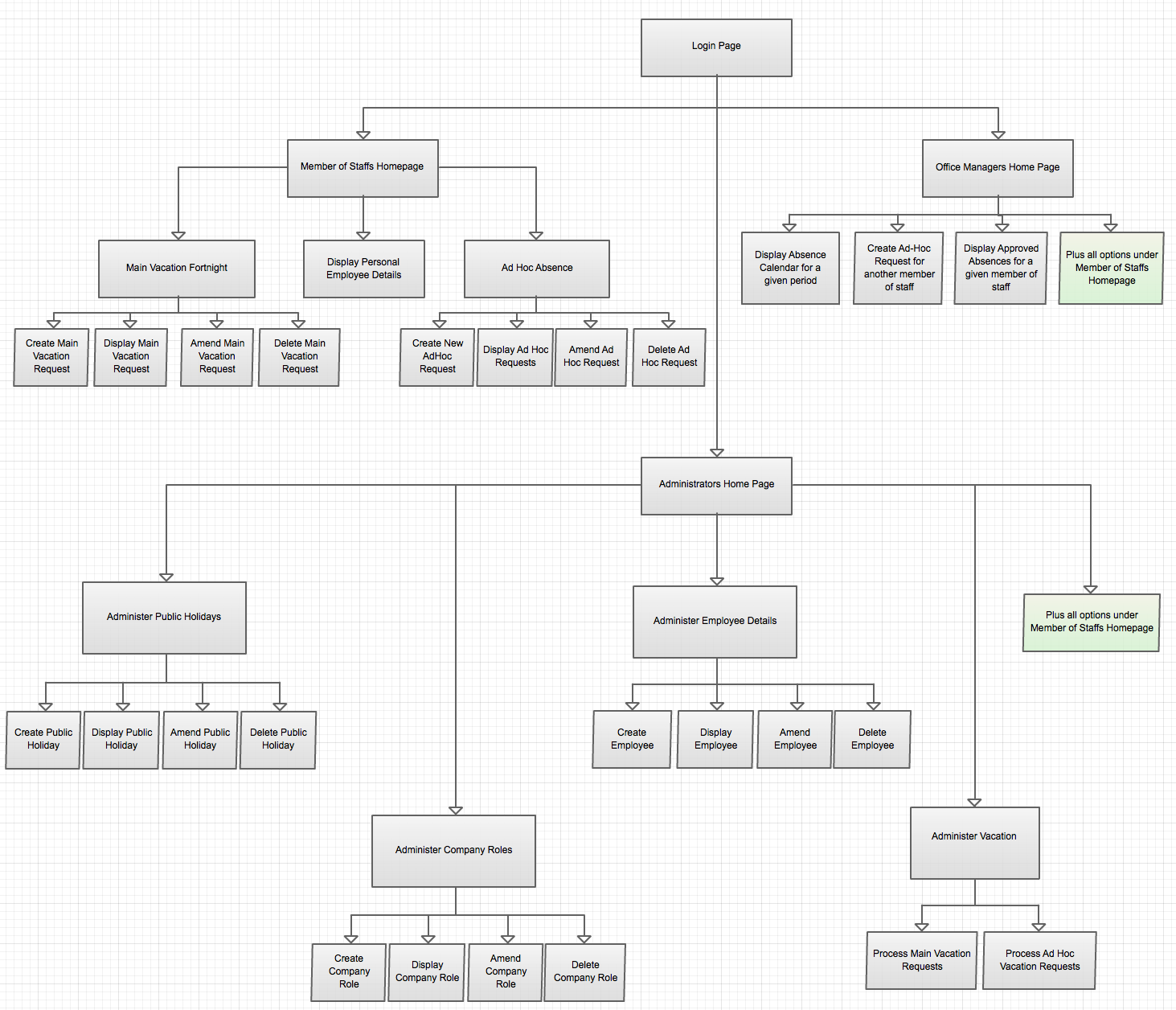
### Introduction

As part of the system development, I need to define a strong test plan to ensure that the system functions as designed and contains no defects. My test plan will use a variety of test techniques to achieve this.

### Top Down Based Approach

My application is based around a series of web pages, in a tree hierarchy as shown below.

My approach to testing will be initially to create a stub for each and every page, with allowing a top down approach to page by page testing, as each page is introduced.



### Trace Tables

For some of the key and most complex functionality, described earlier in pseudocode, I will dry run my code using trace tables prior to any unit testing. By using this approach, I aim to identify any logic errors as early as possible.

### Unit Testing

I will use a white box testing approach for unit testing, using my knowledge of the implementation of each page to exercise each logic path of the code.

Within the unit testing I will include validation and bounds testing of all inputs.

### System and Integration Testing

I will use a black box testing approach for overall system testing, basing the testing on a number of use case scenarios to exercise the functionality of the overall system, ensuring that the resultant output of these use cases matches the expected results.

### Security Testing

Security testing will be conducted to ensure that the minimum password standards described earlier are enforced, to ensure that the users session timeout is enforced, and to ensure that no unencrypted password information is maintained in the database.

# technical solution

Function LoginPageInit.

Function LoginPageSubmit.

Function MainVacationPageInit.

Function MainVacationPageSubmit.

Function function MainVacationPageDelete.

Function EmployeeDetailsInit.

Function AdHocAbsenceInit.

Function AdHocAbsenceSubmit.

Function AdHocAbsenceAmend.

Function AdHocAbsenceDelete.

Function OMApprovedAbsenceInit.

Function OMApprovedAbsenceSearch.

Function OMCreateAbsenceInit.

Function OMCreateAbsenceSubmit.

Function AdminPublicHolidayInit.

Function AdminPublicHolidayCreate.

Function AdminPublicHolidayAmend.

Function AdminPublicHolidayDelete.

Function AdminRoleInit.

Function AdminRoleCreate.

Function AdminRoleAmend.

Function AdminRoleDelete.

Function AdminEmployeeInit.

Function AdminEmployeeCreate.

Function AdminEmployeeAmend.

Function AdminEmployeeDelete.

Function AdminMainVacationInit.

Function AdminMainVacationProcess.

Function AdminAdHocApprove.

Function AdminAdHocReject.

# system testing

# system maintenance

# USER manual

# appraisal

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **COMP4 Commentary Sheet** | | | | | |
| **CENTRE No.** |  |  |  |  |  |
| **CANDIDATE No.** |  |  |  |  |  |
| **Programming Language:** |  | PHP / Javascript / MySQL / CSS / HTML5 |  |  |  |
| **Title / Type of System** |  | CATS – Company Absence Tracking System Database System with Web based interface |  |  |  |
| **Analysis** | **Band** | **Comments** | **Page** | **Mark** | **Max** |
| Background & Problem identification |  |  | 3 |  |  |
| Description of current system |  |  | 3 |  |  |
| User(s) identification |  |  | 7 |  |  |
| User needs & limitations |  |  | 8 |  |  |
| Data source(s) and destination(s) |  |  | 9 |  |  |
| Data volumes |  |  | 10 |  |  |
| Analysis data dictionary |  |  | 10 |  |  |
| DFD's existing & proposed |  |  | 11 |  |  |
| Objectives |  |  | 13 |  |  |
| Complexity |  |  | 14 |  |  |
| Potential solutions |  |  | 15 |  |  |
| Proposed solution |  |  | 16 |  |  |
| Use of formal method(s) |  |  | 16 |  |  |
| E-R model / Objects etc. (if appropriate) | |  | 17 |  | **12** |
| **Design** |  |  |  |  |  |
| Overall system design |  |  | 19 |  |  |
| Modular structure |  |  | 20 |  |  |
| Record or d'base structure |  |  | 21 |  |  |
| Validation incl errors |  |  |  |  |  |
| File org & processing |  |  | 29 |  |  |
| D'base design + E-R model |  |  | 29 |  |  |
| Storage media & format |  |  |  |  |  |
| Algs for data transformation |  |  |  |  |  |
| User interface (I/O) |  |  |  |  |  |
| Security & integrity of data |  |  |  |  |  |
| System security |  |  |  |  |  |
| Test strategy |  |  |  |  | **12** |
| **Technical Soln** |  |  |  |  |  |
| Technical competence |  |  |  |  |  |
| Complex tasks |  |  |  |  |  |
| Annotated "listing" |  |  |  |  |  |
| Samples of screens and design views |  |  |  |  |  |
| Customisation |  |  |  |  | **20** |
| **System Testing** |  | **THIS SECTION IS NOT RELATED TO COMPLEXITY** |  |  |  |
| Design of test plan |  |  |  |  |  |
| Minimal test data cross referenced |  |  |  |  |  |
| Typical data |  |  |  |  |  |
| Erroneous data |  |  |  |  |  |
| Extreme (boundary) data |  |  |  |  |  |
| Annotated results for above |  |  |  |  | **8** |
| **System Maintenance** |  |  |  |  |  |
| System overview |  |  |  |  |  |
| Algorithms or alternative |  |  |  |  |  |
| Annotated listings/screens |  |  |  |  |  |
| Proc & VAR lists (prog) |  |  |  |  |  |
| Forms, macros etc (package) |  |  |  |  | **7** |
| **User Manual** |  | **A FULL USER MANUAL IS NOW REQUIRED, NOT JUST ONE SECTION OF THE SYSTEM** | | | |
| Contents page |  |  |  |  |  |
| Introduction & how to install |  |  |  |  |  |
| Screen displays explained |  |  |  |  |  |
| Error handling |  |  |  |  |  |
| Appropriate level for user |  |  |  |  |  |
| Enables Easy Use of System |  |  |  |  | **7** |
| **Appraisal** |  |  |  |  |  |
| Objectives evaluated |  |  |  |  |  |
| Further development? |  |  |  |  |  |
| User feedback & analysis of this |  |  |  |  | **6** |
| **Quality of Language** |  |  |  |  | **3** |
| **Total** |  |  |  |  | **75** |
| **N.B. Some of these sections will NOT be appropriate for certain types of project, especially non-Data Processing ones. Not all of them may need to be completed. In all cases, you MUST refer to the Full Assessment Criteria if using this grid.** | | | | | |