**ISAD251 Report**

APEX811: Cody Robinson, Reece Davies,

Jaime Kershaw Brown, Erin Clifton

**Plymouth University**

**Nov 2018**

Contents

[Introduction 3](#_Toc531866091)

[1. Scenario 3](#_Toc531866092)

[2. Background information 3](#_Toc531866093)

[3. Assumptions made 3](#_Toc531866094)

[4. Functional requirements 4](#_Toc531866095)

[5. Project plan 5](#_Toc531866096)

[Development 6](#_Toc531866097)

[1. Method of Approach 6](#_Toc531866098)

[2. Entity Relationship Diagram 7](#_Toc531866099)

[3. SQL Scripts 8](#_Toc531866100)

[4. Student App Development Screenshots 13](#_Toc531866101)

[5. Admin App Development Screenshots 16](#_Toc531866102)

[Critical Appraisal 25](#_Toc531866103)

# Introduction

## Scenario

There is a need for a university system that allows stage-two computing students to apply for a placement year. This application will be tracking the student’s activity throughout the application process, from the start of stage-two until they have secured a placement. It will not focus solely on students, but also be capable of managing companies and the jobs available to students. Students will be capable of applying through a mobile app, whilst the university will be able to manage students and companies through the admin desktop app.

## Background information

Relating specifically to the requirements of the system, the admin will be required to input the students’ information in order for them to use the system. This includes: student ID (which is auto generated), the code of the programme they are undertaking, full name, email address, mobile and landline telephone numbers, date of birth, home and term time address. Furthermore, specific values must not be null and in the format that best suites the content. In order for a student to apply for a job, they are required to submit a CV and have it approved by the admin. Once this has been accomplished, they will be deemed ‘active’ and can start applying for jobs.

A company is capable of having one or more sites; the jobs provided by a company will be related to a specific site the company owns, and thus will be the location for the job. This includes the address and postcode. In order for the student to apply for a job, they must be informed about the specifics of the job, and therefore the job must include: a job title, a short description outlining the job, contact information for enquiries, the salary of the job, the start date for the placement, a closing date for applications and an application method. Specific jobs will also require information relating to the student’s application history, which will outline the dates and times of the student’s status throughout their previous applications. The admin will be responsible for adding companies, sites and jobs.

## Assumptions made

* Each site will have its own independent email address and telephone number for students to contact enquiring about the job available.
* Each site can have multiple jobs.
* Each job can have multiple vacancies available.
* Each company should have at least one site.
* There is no limit for how many sites a company can have.
* Admin will be responsible for creating the students’ accounts.
* A company’s name will not change.
* Once a student has accepted a placement they shall make no further applications.
* Companies will not close down sites during the application process.
* The company will make their choice of applicant for a job after the closing date.
* A student will not make multiple applications to the same job.
* A student will not be entered into the system twice.
* A student can only be at each stage in the application process once

## Functional requirements

**Admin Desktop Application**

**Students**

- Add a student to the database

- View a student’s information

- View lists of both active & inactive students

- View a list of active students who have not yet accepted an offer

- View a list of students who have accepted a placement offer

- View a report of a student’s application(s) status

- View a pie chart showing the proportion of placed to unplaced students

- View a stacked bar chart showing the number of active students placed & unplaced by programme

- View a calendar of job vacancy closing dates

- Amend student information

**Companies**

- Add a company to the database

- View a company’s information

- Update a company’s information

**Job Vacancies**

- Add a job vacancy

- View an individual job

- View a list of jobs (current & past job vacancies)

- View a list of jobs that close within the next 7 days

- Update a given job vacancy

**Student Mobile Application**

**Jobs**

- View a list of current jobs (current & past job vacancies)

**Applications**

- Record an application made

- Add a new status of application

- View a report on the status of all applications and each application made

**Students**

- View a student’s information including

- Amend student information

## Project plan

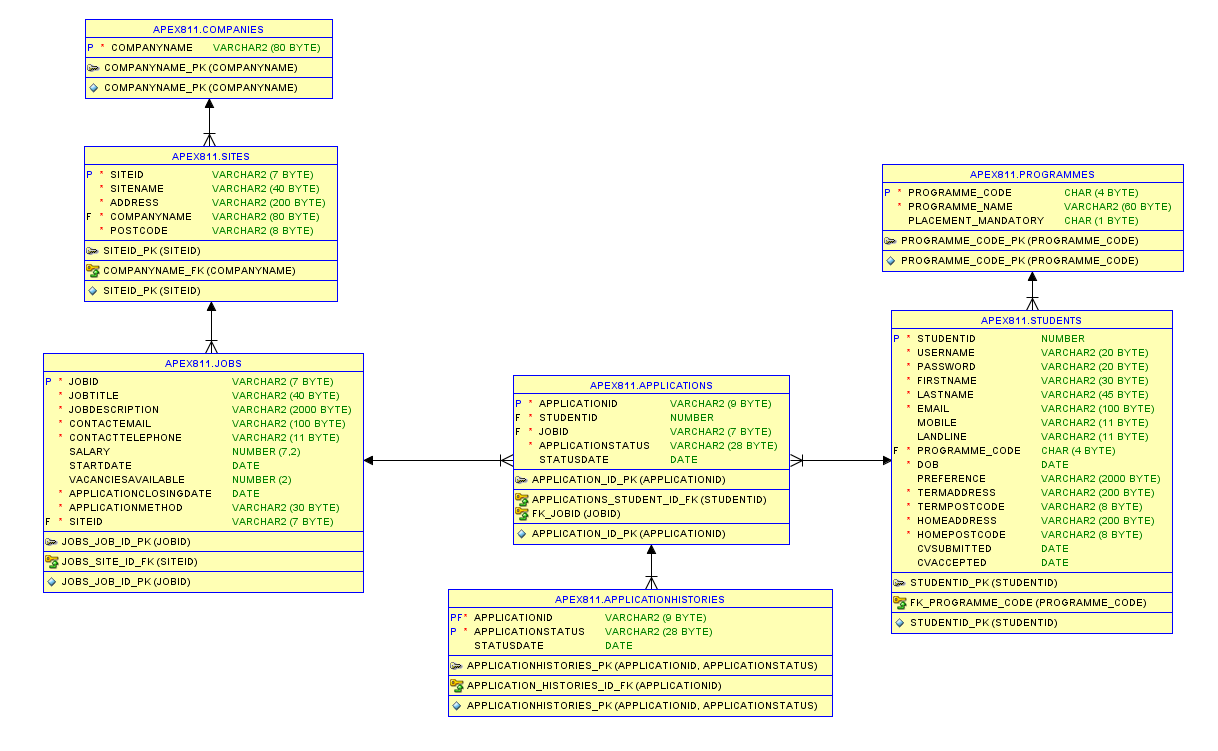
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Activity | Expected Start Date | Estimate Hours (Per Person) | Actual Start date | Actual Hours spent (Per Person) |
| Read through of Assignment brief, and discussion. | 22/10/2018 | 2 | 18/10/2018 | 2 |
| Create Project Plan for assignment. | 24/10/2018 | 5 | 31/10/2018 | 4 |
| Listed all the functional requirements. | 26/10/2018 | 2 | 23/10/2018 | 5 |
| List all attributes. | 29/10/2018 | 1 | 19/10/2018 | 1 |
| Normalise attributes into third normal form, alongside with declaring their table name and their data types. | 31/10/2018 | 6 | 19/10/2018 | 8 |
| Create relevant ERD for normalised data. | 06/11/2018 | 4 | 22/10/2018 | 3 |
| Design storyboard for front-end view of system. | 09/11/2018 | 3 | 31/10/2018 | 6 |
| Create SQL scripts to generate all tables in Oracle, alongside with constraints to relevant primary, composite and foreign keys. | 14/11/2018 | 5 | 29/10/2018 | 7 |
| Popularised the database with relevant information. | 23/11/2018 | 4 | 26/11/2018 | 4 |
| Create graphical user interface for front-view. | 27/11/2018 | 9 | 27/11/2018 | 17 |
| Create Report to review Assignment. | 07/12/2018 | 9 | 03/12/2018 | 12 |

# Development

## Method of Approach

Once we had our four members for the group, we arranged a meeting in which we read through the assignment brief and discussed which tasks should be performed first with the best method to do so. Following this we started naming all the attributes that the system might contain, including the ones stated in the assignment brief. This may have not been the best approach as we did not initially create a project plan prior to starting the coursework. As a result, we were required to spend more time planning on what actions were left to do. This is because we were unsure of which tasks were complete, semi-complete or incomplete. Once the project plan was created, it acted as a foundation of what task must be worked upon, when it must be started, and the time spent on that specific task. Of course, certain tasks had complications and thus the actual amount of hours spent on it exceeded the estimate amount of hours. This was particularly problematic when developing the graphical user interface with APEX as we were unfamiliar with the software.

## Entity Relationship Diagram



## SQL Scripts

CREATE TABLE "APEX811"."COMPANIES"

( "COMPANYNAME" VARCHAR2(80 BYTE) CONSTRAINT "COMPANYNAME\_NN" NOT NULL ENABLE,

CONSTRAINT "COMPANYNAME\_PK" PRIMARY KEY ("COMPANYNAME")

);

CREATE TABLE "APEX811"."SITES"

( "SITEID" VARCHAR2(7 BYTE) DEFAULT ON NULL 'S'||LTRIM(TO\_CHAR("APEX811"."SEQ\_SITE\_ID"."NEXTVAL",'09999')) NOT NULL ENABLE,

"SITENAME" VARCHAR2(40 BYTE) CONSTRAINT "SITENAME\_NN" NOT NULL ENABLE,

"ADDRESS" VARCHAR2(200 BYTE) CONSTRAINT "ADDRESS\_NN" NOT NULL ENABLE,

"COMPANYNAME" VARCHAR2(80 BYTE) CONSTRAINT "SITE\_COMPANYNAME\_NN" NOT NULL ENABLE,

"POSTCODE" VARCHAR2(8 BYTE) CONSTRAINT "SITE\_POSTCODE\_NN" NOT NULL ENABLE,

CONSTRAINT "SITEID\_PK" PRIMARY KEY ("SITEID"),

CONSTRAINT "SITE\_POSTCODE\_CHK" CHECK (postcode LIKE '%\_\_\_ \_\_\_') ENABLE,

CONSTRAINT "COMPANYNAME\_FK" FOREIGN KEY ("COMPANYNAME")

REFERENCES "APEX811"."COMPANIES" ("COMPANYNAME") ENABLE

) ;

CREATE SEQUENCE "APEX811"."SEQ\_SITE\_ID" MINVALUE 1 MAXVALUE 99999 INCREMENT BY 1 START WITH 10016 NOCACHE NOORDER CYCLE NOKEEP NOSCALE GLOBAL ;

CREATE TABLE "APEX811"."JOBS"

( "JOBID" VARCHAR2(7 BYTE) DEFAULT ON NULL 'J'||LTRIM(TO\_CHAR("APEX811"."SEQ\_JOB\_ID"."NEXTVAL",'09999')) NOT NULL ENABLE,

"JOBTITLE" VARCHAR2(40 BYTE) CONSTRAINT "JOBS\_JOB\_TITLE\_NN" NOT NULL ENABLE,

"JOBDESCRIPTION" VARCHAR2(2000 BYTE) CONSTRAINT "JOBS\_JOB\_DESCRIPTION\_NN" NOT NULL ENABLE,

"CONTACTEMAIL" VARCHAR2(100 BYTE) CONSTRAINT "JOB\_EMAIL\_NN" NOT NULL ENABLE,

"CONTACTTELEPHONE" VARCHAR2(11 BYTE) CONSTRAINT "JOB\_TELEPHONE\_NN" NOT NULL ENABLE,

"SALARY" NUMBER(7,2),

"STARTDATE" DATE,

"VACANCIESAVAILABLE" NUMBER(2,0),

"APPLICATIONCLOSINGDATE" DATE CONSTRAINT "JOBS\_APPLICATION\_CLOSING\_DATE\_NN" NOT NULL ENABLE,

"APPLICATIONMETHOD" VARCHAR2(30 BYTE) CONSTRAINT "JOBS\_APPLICATION\_METHOD\_NN" NOT NULL ENABLE,

"SITEID" VARCHAR2(7 BYTE) CONSTRAINT "SITEID\_NN" NOT NULL ENABLE,

CONSTRAINT "JOBS\_JOB\_ID\_PK" PRIMARY KEY ("JOBID"),

CONSTRAINT "JOB\_EMAIL\_CHK" CHECK (contactemail LIKE '%\_@\_\_\_%.\_\_%') ENABLE,

CONSTRAINT "JOB\_MOBILE\_CHK" CHECK (contacttelephone NOT LIKE '%[^0-9]%') ENABLE,

CONSTRAINT "JOBS\_SITE\_ID\_FK" FOREIGN KEY ("SITEID")

REFERENCES "APEX811"."SITES" ("SITEID") ENABLE

) ;

CREATE SEQUENCE "APEX811"."SEQ\_JOB\_ID" MINVALUE 1 MAXVALUE 99999 INCREMENT BY 1 START WITH 10024 NOCACHE NOORDER CYCLE NOKEEP NOSCALE GLOBAL ;

CREATE TABLE "APEX811"."PROGRAMMES"

( "PROGRAMME\_CODE" CHAR(4 BYTE),

"PROGRAMME\_NAME" VARCHAR2(60 BYTE) CONSTRAINT "PROGRAME\_NAME\_NN" NOT NULL ENABLE,

"PLACEMENT\_MANDATORY" CHAR(1 BYTE),

CONSTRAINT "PROGRAMME\_CODE\_PK" PRIMARY KEY ("PROGRAMME\_CODE")

) ;

CREATE TABLE "APEX811"."STUDENTS"

( "STUDENTID" NUMBER DEFAULT ON NULL "APEX811"."SEQ\_STUDENT\_ID"."NEXTVAL" NOT NULL ENABLE,

"USERNAME" VARCHAR2(20 BYTE) CONSTRAINT "USERNAME\_NN" NOT NULL ENABLE,

"PASSWORD" VARCHAR2(20 BYTE) CONSTRAINT "PASSWORD\_NN" NOT NULL ENABLE,

"FIRSTNAME" VARCHAR2(30 BYTE) CONSTRAINT "FIRSTNAME\_NN" NOT NULL ENABLE,

"LASTNAME" VARCHAR2(45 BYTE) CONSTRAINT "LASTNAME\_NN" NOT NULL ENABLE,

"EMAIL" VARCHAR2(100 BYTE) CONSTRAINT "STUDENT\_EMAIL\_NN" NOT NULL ENABLE,

"MOBILE" VARCHAR2(11 BYTE),

"LANDLINE" VARCHAR2(11 BYTE),

"PROGRAMME\_CODE" CHAR(4 BYTE) CONSTRAINT "PROGRAMME\_CODE\_NN" NOT NULL ENABLE,

"DOB" DATE CONSTRAINT "DOB\_NN" NOT NULL ENABLE,

"PREFERENCE" VARCHAR2(2000 BYTE),

"TERMADDRESS" VARCHAR2(200 BYTE) CONSTRAINT "STUDENT\_TERMADDRESS\_NN" NOT NULL ENABLE,

"TERMPOSTCODE" VARCHAR2(8 BYTE) CONSTRAINT "STUDENT\_TERMPOSTCODE\_NN" NOT NULL ENABLE,

"HOMEADDRESS" VARCHAR2(200 BYTE) CONSTRAINT "STUDENT\_HOMEADDRESS\_NN" NOT NULL ENABLE,

"HOMEPOSTCODE" VARCHAR2(8 BYTE) CONSTRAINT "STUDENT\_HOMEPOSTCODE\_NN" NOT NULL ENABLE,

"CVSUBMITTED" DATE,

"CVACCEPTED" DATE,

CONSTRAINT "STUDENTID\_PK" PRIMARY KEY ("STUDENTID"),

CONSTRAINT "STUDENT\_EMAIL\_CHK" CHECK (email LIKE '%\_@\_\_\_%.\_\_%') ENABLE,

CONSTRAINT "STUDENT\_MOBILE\_CHK" CHECK (mobile NOT LIKE '%[^0-9]%') ENABLE,

CONSTRAINT "STUDENT\_LANDLINE\_CHK" CHECK (landline NOT LIKE '%[^0-9]%') ENABLE,

CONSTRAINT "STUDENT\_TERMPOSTCODE\_CHK" CHECK (termpostcode LIKE '%\_\_\_ \_\_\_') ENABLE,

CONSTRAINT "STUDENT\_HOMEPOSTCODE\_CHK" CHECK (homepostcode LIKE '%\_\_\_ \_\_\_') ENABLE,

CONSTRAINT "FK\_PROGRAMME\_CODE" FOREIGN KEY ("PROGRAMME\_CODE")

REFERENCES "APEX811"."PROGRAMMES" ("PROGRAMME\_CODE") ENABLE

) ;

CREATE SEQUENCE "APEX811"."SEQ\_STUDENT\_ID" MINVALUE 1 MAXVALUE 999999 INCREMENT BY 1 START WITH 100042 NOCACHE NOORDER CYCLE NOKEEP NOSCALE GLOBAL ;

CREATE OR REPLACE EDITIONABLE TRIGGER "APEX811"."TRG\_CV\_ACCEPTED"

BEFORE UPDATE OF CVACCEPTED ON STUDENTS FOR EACH ROW

BEGIN

IF :OLD.CVSUBMITTED IS NULL

THEN :NEW.CVACCEPTED := NULL;

END IF;

END;

ALTER TRIGGER "APEX811"."TRG\_CV\_ACCEPTED" ENABLE;

CREATE OR REPLACE EDITIONABLE TRIGGER "APEX811"."TRG\_DOB"

BEFORE INSERT OR UPDATE OF DOB ON STUDENTS FOR EACH ROW

BEGIN

IF (:NEW.dob + NUMTOYMINTERVAL(18,'YEAR')) >

SYSDATE THEN

RAISE\_APPLICATION\_ERROR(-18, 'Student must be at least 18 years old');

END IF;

END;

ALTER TRIGGER "APEX811"."TRG\_DOB" ENABLE;

CREATE TABLE "APEX811"."APPLICATIONS"

( "APPLICATIONID" VARCHAR2(9 BYTE) DEFAULT ON NULL 'A'||LTRIM(TO\_CHAR("APEX811"."SEQ\_APPLICATION\_ID"."NEXTVAL",'0999999')) NOT NULL ENABLE,

"STUDENTID" NUMBER CONSTRAINT "STUDENT\_ID\_NN" NOT NULL ENABLE,

"JOBID" VARCHAR2(7 BYTE) CONSTRAINT "JOB\_ID\_NN" NOT NULL ENABLE,

"APPLICATIONSTATUS" VARCHAR2(28 BYTE) CONSTRAINT "APPLICATIONSTATUS\_NN" NOT NULL ENABLE,

"STATUSDATE" DATE,

CONSTRAINT "APPLICATION\_ID\_PK" PRIMARY KEY ("APPLICATIONID"),

CONSTRAINT "APPLICATIONS\_STUDENT\_ID\_FK" FOREIGN KEY ("STUDENTID")

REFERENCES "APEX811"."STUDENTS" ("STUDENTID") ENABLE,

CONSTRAINT "FK\_JOBID" FOREIGN KEY ("JOBID")

REFERENCES "APEX811"."JOBS" ("JOBID") ENABLE

) ;

CREATE SEQUENCE "APEX811"."SEQ\_APPLICATION\_ID" MINVALUE 1 MAXVALUE 9999999 INCREMENT BY 1 START WITH 1000029 NOCACHE NOORDER CYCLE NOKEEP NOSCALE GLOBAL ;

CREATE OR REPLACE TRIGGER TRG\_APPLICATION\_UPDATE

BEFORE UPDATE OF APPLICATIONSTATUS, STATUSDATE ON APPLICATIONS FOR EACH ROW

DECLARE

HISTORY\_STATUS VARCHAR(40);

HISTORY\_STATUS\_COUNT NUMBER;

BEGIN

SELECT COUNT(APPLICATIONSTATUS) INTO HISTORY\_STATUS\_COUNT

FROM APPLICATIONHISTORIES

WHERE APPLICATIONHISTORIES.APPLICATIONID = :NEW.APPLICATIONID

AND APPLICATIONHISTORIES.APPLICATIONSTATUS = :NEW.APPLICATIONSTATUS;

IF HISTORY\_STATUS\_COUNT > 0 THEN

SELECT APPLICATIONHISTORIES.APPLICATIONSTATUS INTO HISTORY\_STATUS

FROM APPLICATIONHISTORIES

WHERE APPLICATIONHISTORIES.APPLICATIONID = :NEW.APPLICATIONID

AND APPLICATIONHISTORIES.APPLICATIONSTATUS = :NEW.APPLICATIONSTATUS;

ELSE

HISTORY\_STATUS := NULL;

END IF;

IF :OLD.APPLICATIONSTATUS = :NEW.APPLICATIONSTATUS OR HISTORY\_STATUS = :NEW.APPLICATIONSTATUS OR

THEN

RAISE\_APPLICATION\_ERROR(-20007,'You cannot select the same stage twice');

ELSE

INSERT INTO APPLICATIONHISTORIES

(

APPLICATIONID, APPLICATIONSTATUS, STATUSDATE

)

VALUES

(

:OLD.APPLICATIONID, :OLD.APPLICATIONSTATUS, :OLD.STATUSDATE

);

END IF;

END;

CREATE OR REPLACE TRIGGER CREATE\_APPLICATION\_TRIGGER

BEFORE INSERT OR UPDATE OF STUDENTID ON APPLICATIONS FOR EACH ROW

DECLARE ACCEPTEDCV DATE;

ALREADYAPPLIEDNUMBER NUMBER;

BEGIN

SELECT COUNT(APPLICATIONS.STUDENTID) INTO ALREADYAPPLIEDNUMBER

FROM APPLICATIONS

WHERE APPLICATIONS.STUDENTID = :NEW.STUDENTID AND APPLICATIONS.JOBID = :NEW.JOBID;

IF ALREADYAPPLIEDNUMBER > 0

THEN

RAISE\_APPLICATION\_ERROR(-20008, 'YOU HAVE ALREADY APPLIED TO THIS JOB.');

END IF;

SELECT STUDENTS.CVACCEPTED INTO ACCEPTEDCV

FROM STUDENTS

WHERE STUDENTS.STUDENTID = :NEW.STUDENTID;

IF ACCEPTEDCV IS NULL

THEN

RAISE\_APPLICATION\_ERROR(-20006, 'YOU MUST HAVE AN APPROVED CV.');

END IF;

END;

CREATE TABLE "APEX811"."APPLICATIONHISTORIES"

( "APPLICATIONID" VARCHAR2(9 BYTE),

"APPLICATIONSTATUS" VARCHAR2(28 BYTE) CONSTRAINT "APPLICATIONHISTORIESSTATUS\_NN" NOT NULL ENABLE,

"STATUSDATE" DATE,

CONSTRAINT "APPLICATIONHISTORIES\_PK" PRIMARY KEY ("APPLICATIONID", "APPLICATIONSTATUS"),

CONSTRAINT "APPLICATION\_HISTORIES\_ID\_FK" FOREIGN KEY ("APPLICATIONID")

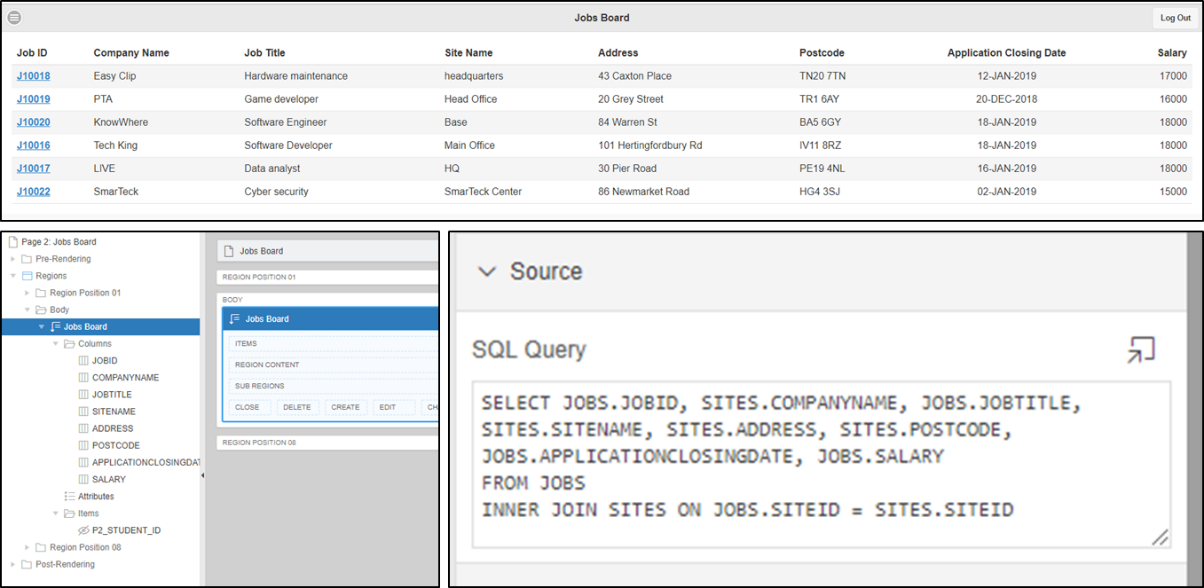
REFERENCES "APEX811"."APPLICATIONS" ("APPLICATIONID") ENABLE

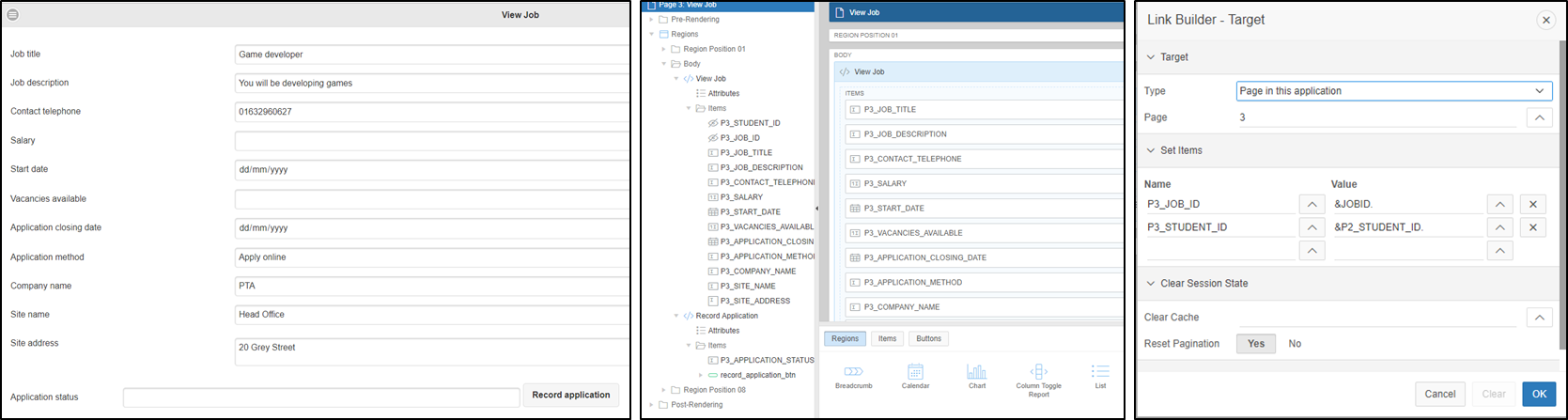
) ;

## Student App Development Screenshots



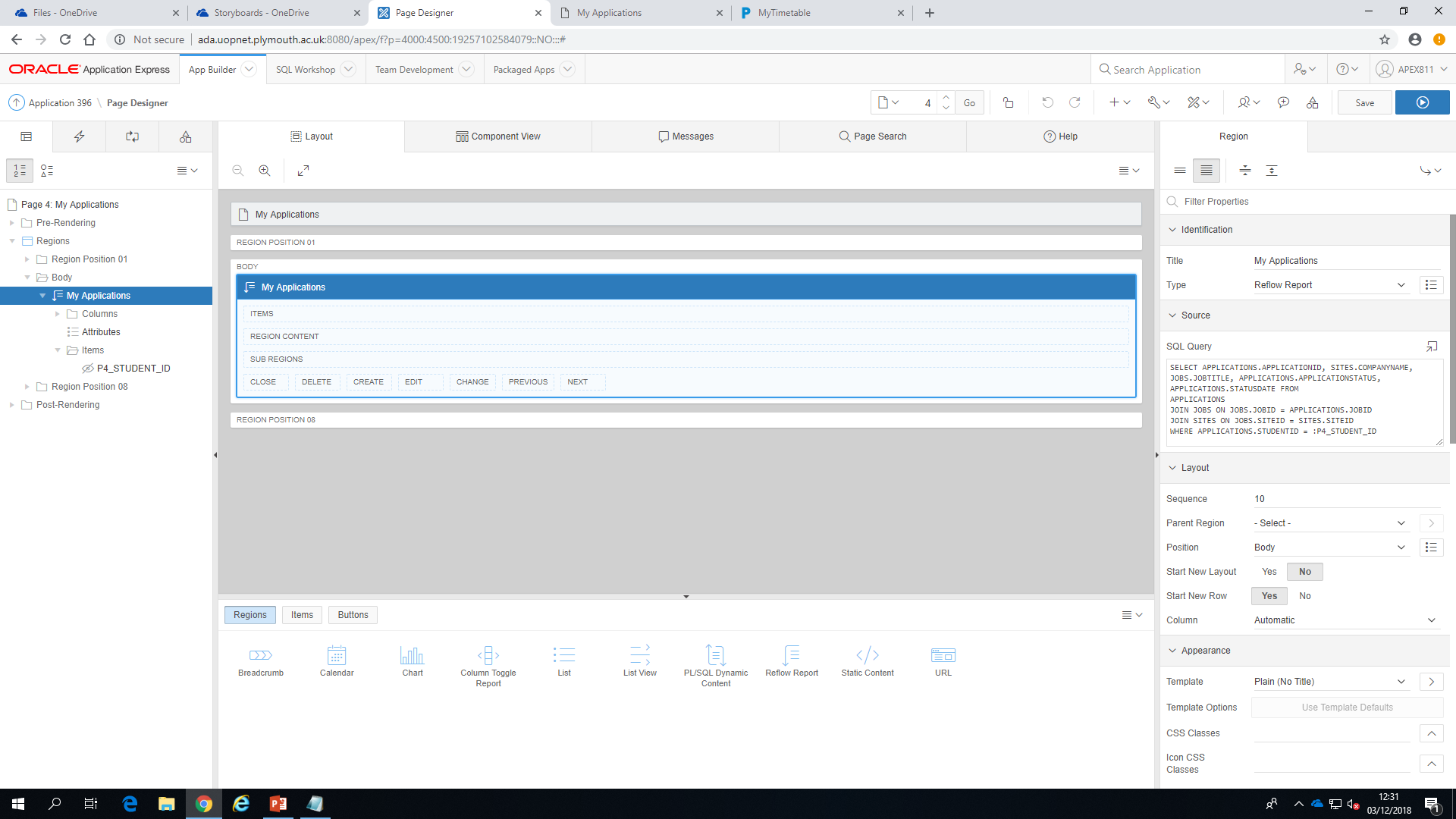
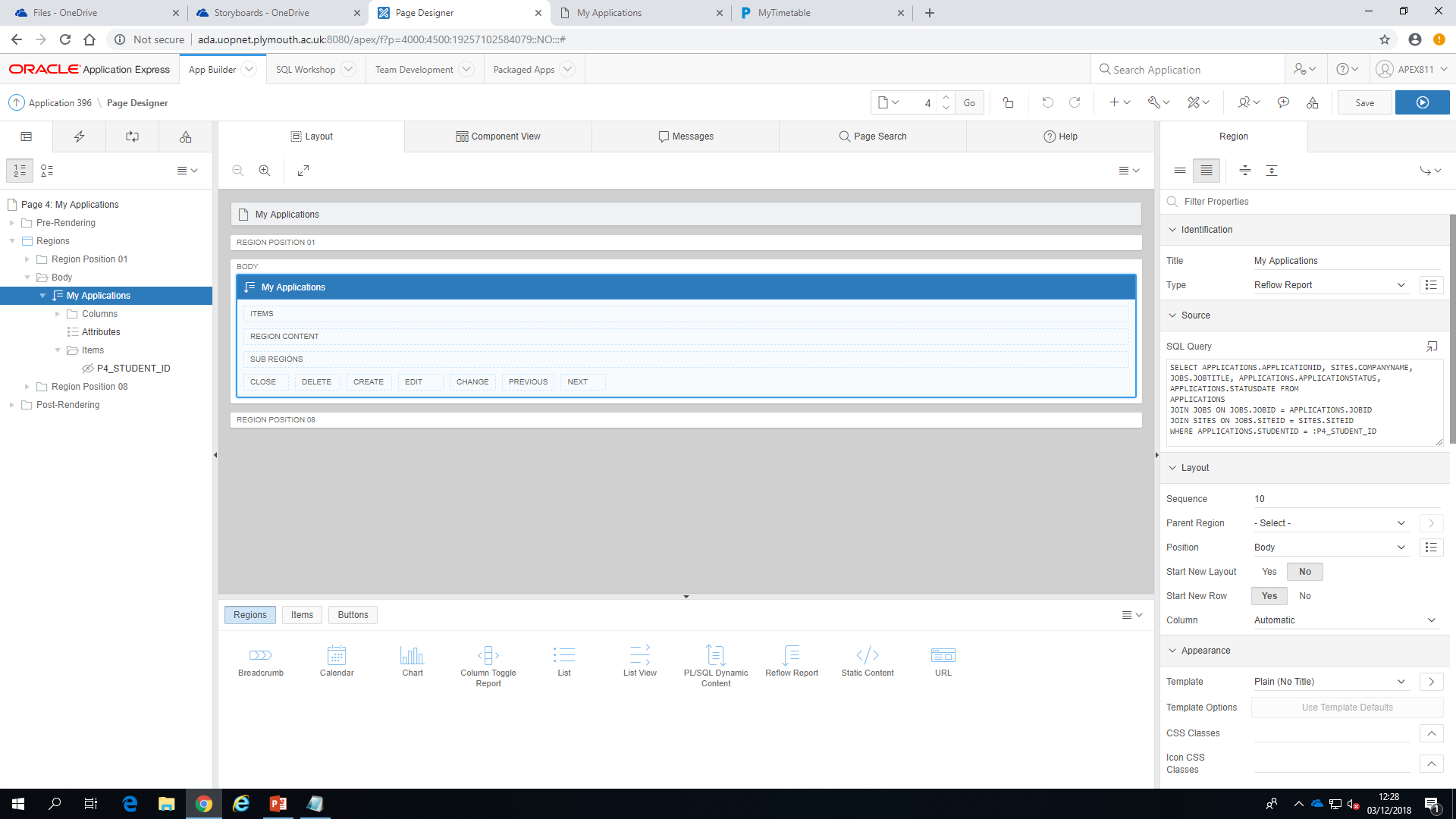
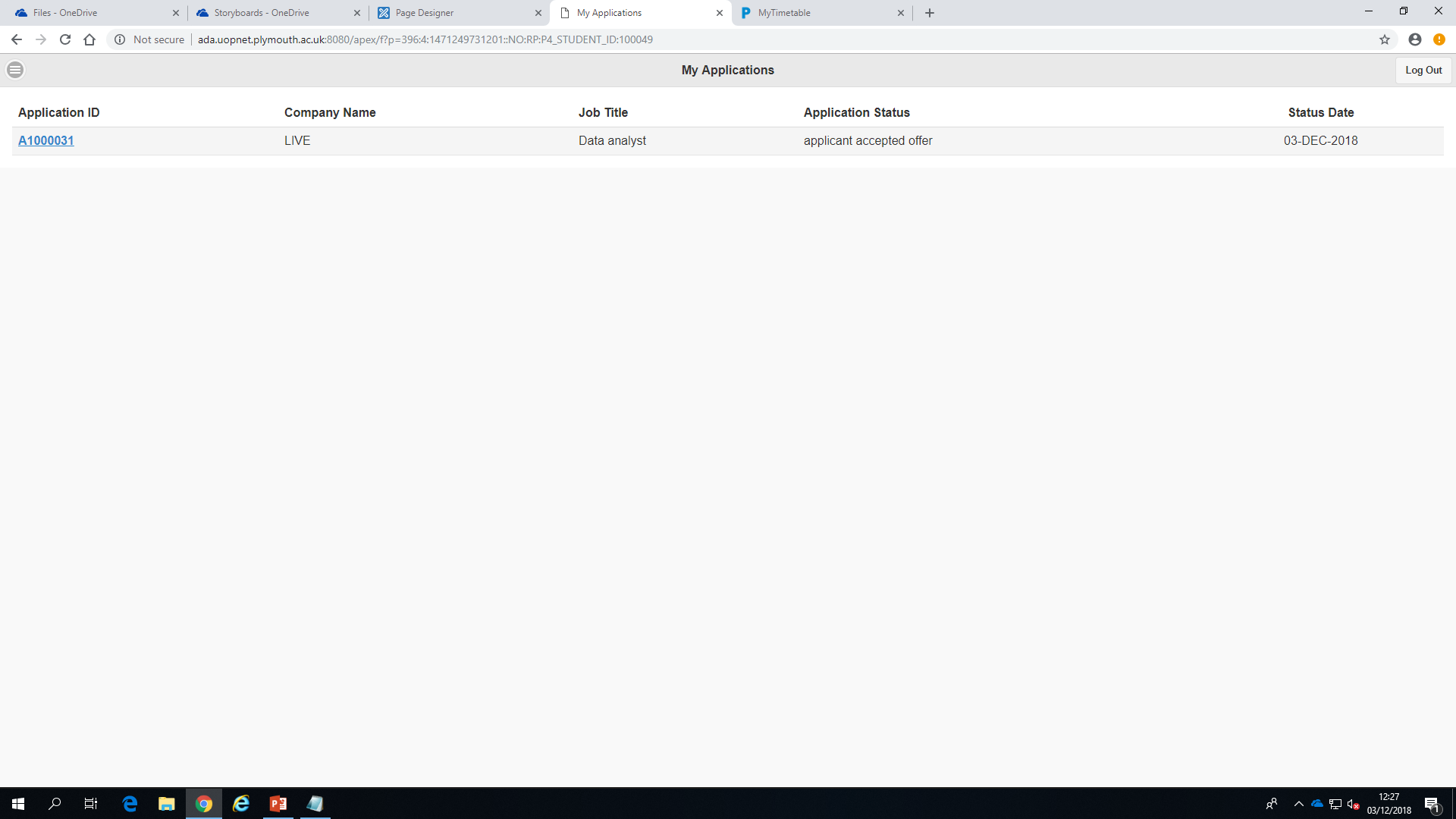
‘Home’ displays navigation buttons to take the user to other pages, which includes ‘Jobs Boards’, ‘My Applications’ and ‘View/Edit My Details’. Also contains a Select List that displays all the students’ names within the drop-down menu.

‘Jobs Board’ contains a reflow report which uses a SQL query to display all relevant information of a job in a table. The user can click on the ‘Job ID’ to be taken to a separate page showing all information specific to the job. Useful for information that does not fit into the table, such as the job description.

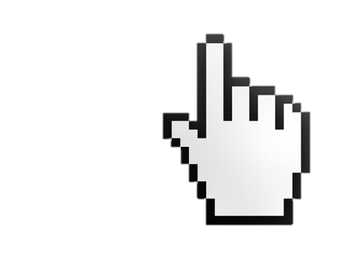
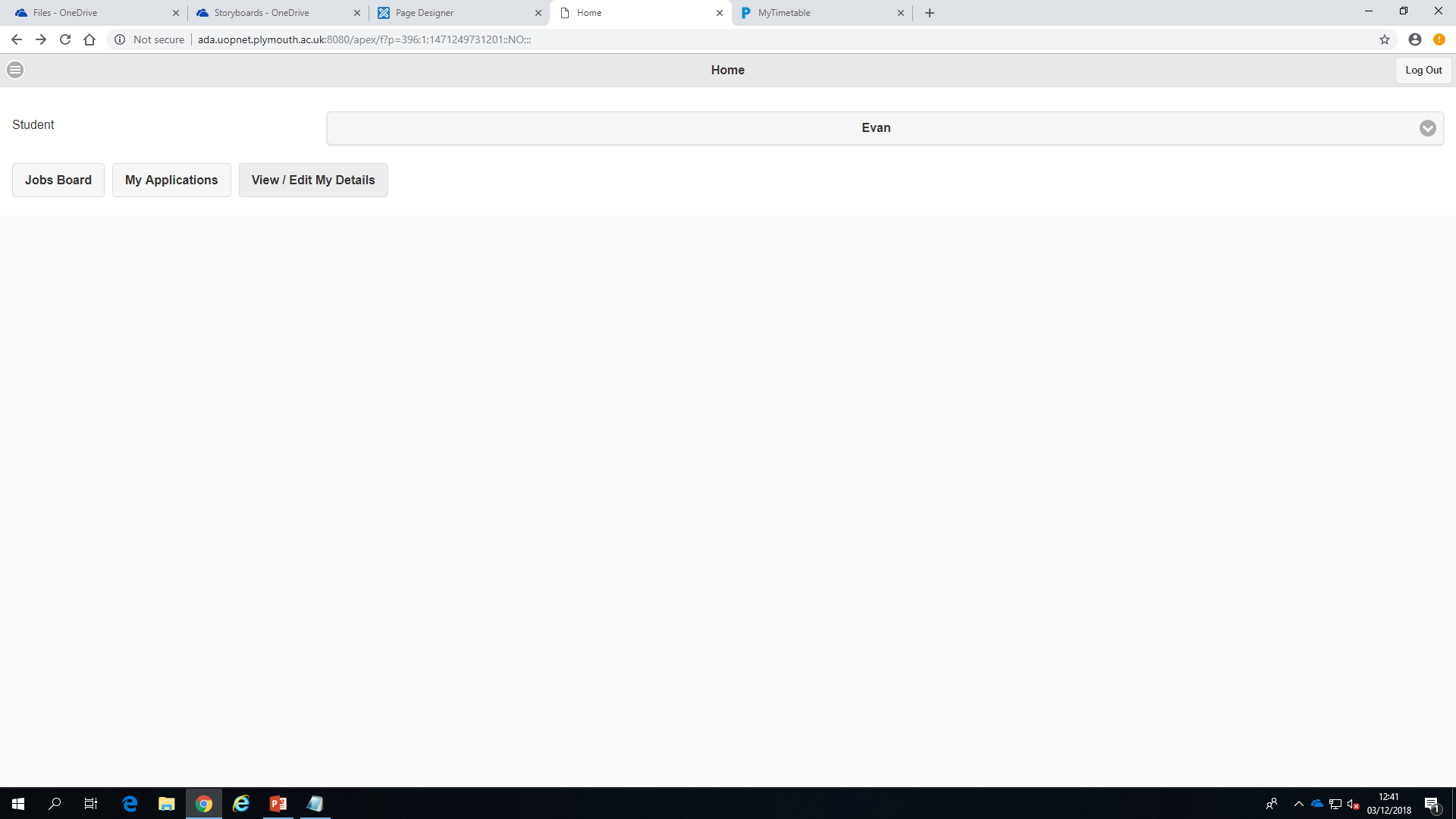
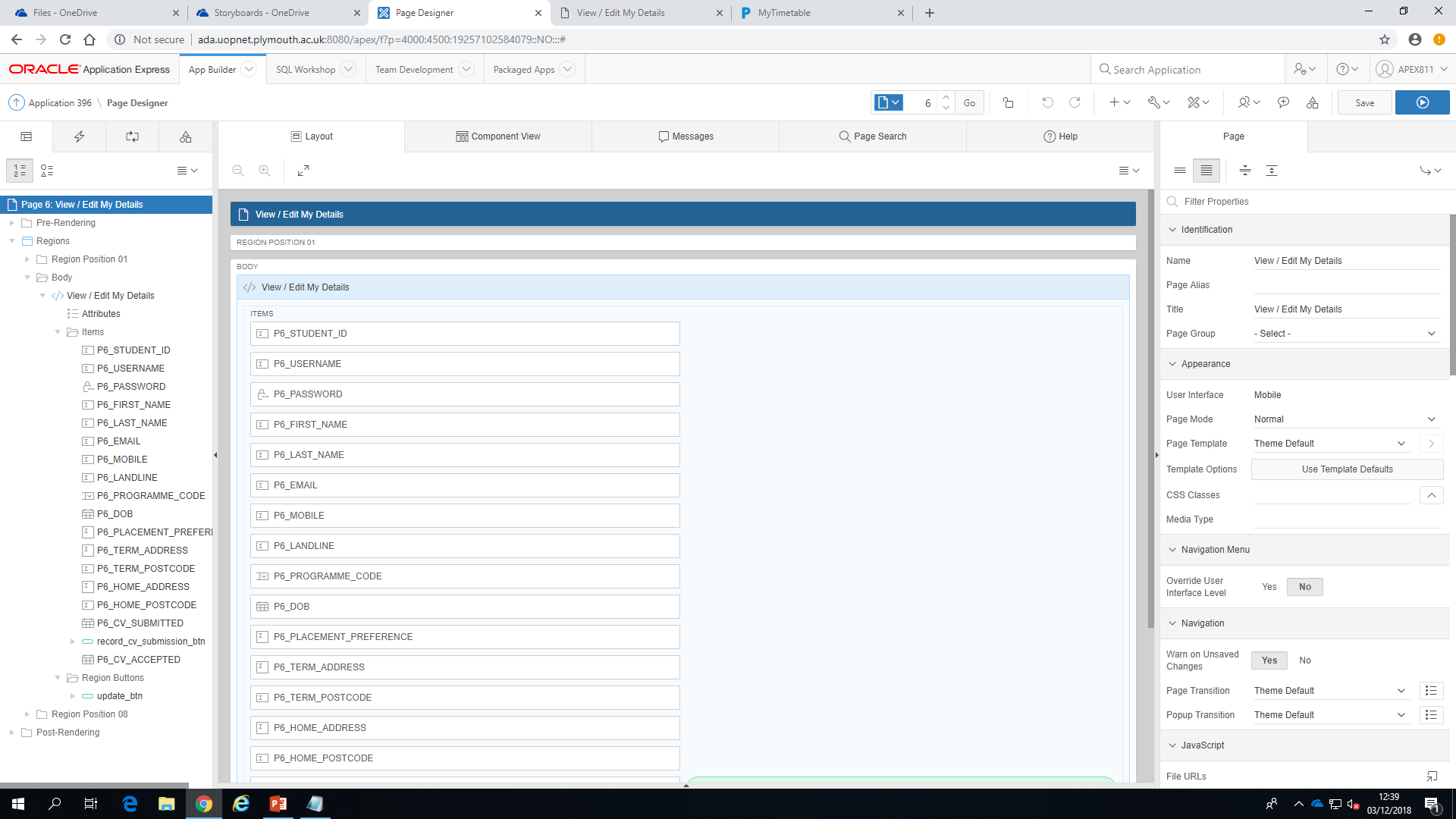
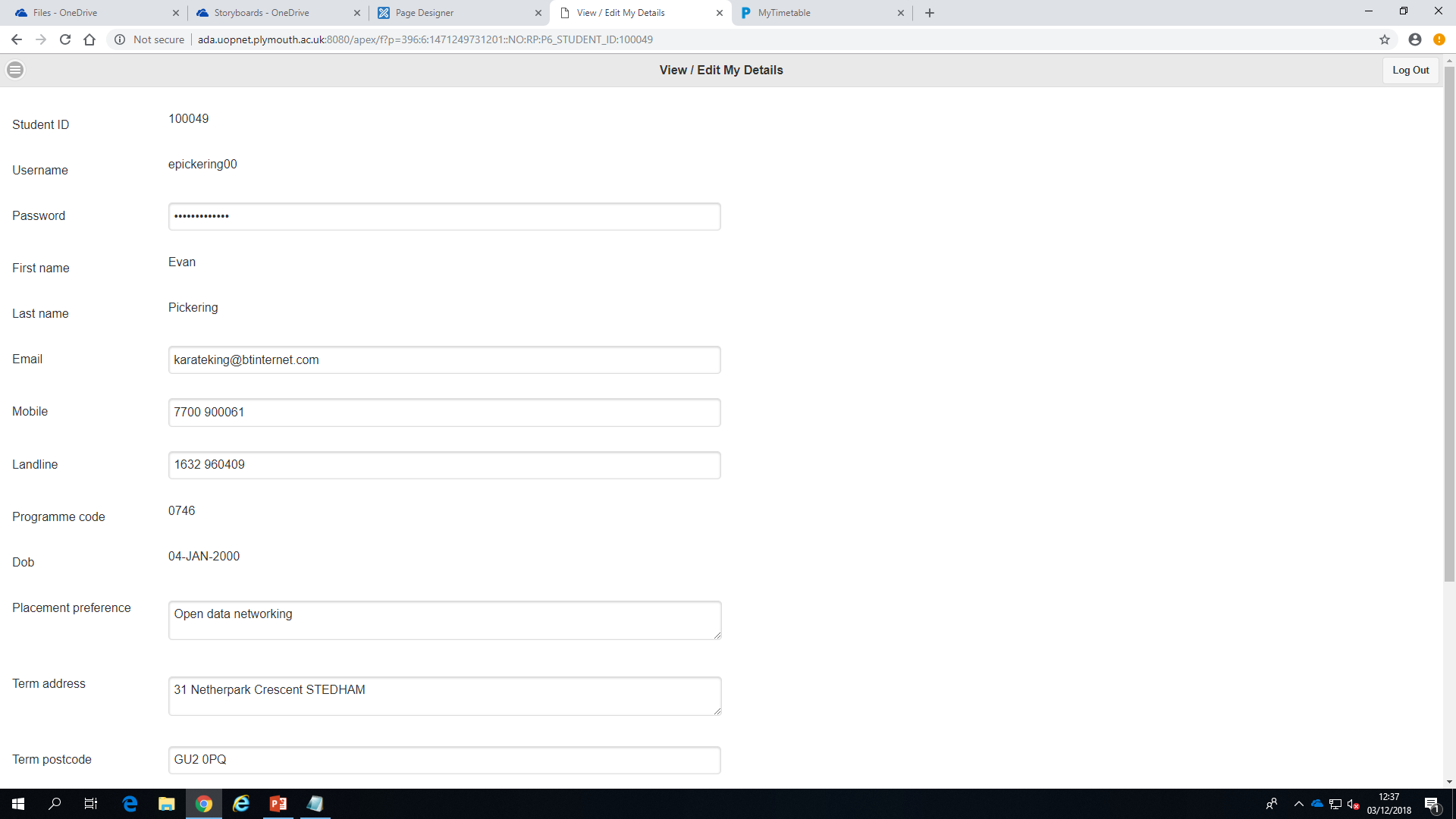


‘View Job’ contains Static Content with columns relevant to the Job’s details. In order to transfer the information across from ‘Jobs Board’ to ‘View Job’, the settings of the column had to be ‘Link’, which will navigate the user to the ‘View Job’ page. The text fields must be automatically filled with the appropriate information when this happens, and therefore we were required to use the ‘Set Name’ property.

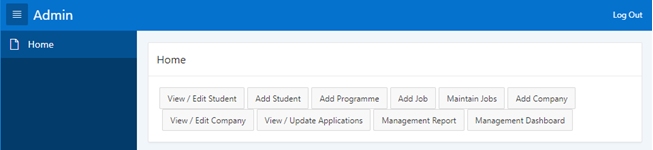
‘My Applications’ is very similar to the ‘Jobs Board’ page, in which it contains a reflow report which extracts specific information from the database and structures it in a table. The relevant headings are ‘Application ID’, ‘Company Name’, ‘Job Title’, ‘Application Status’ and ‘Status Date’.

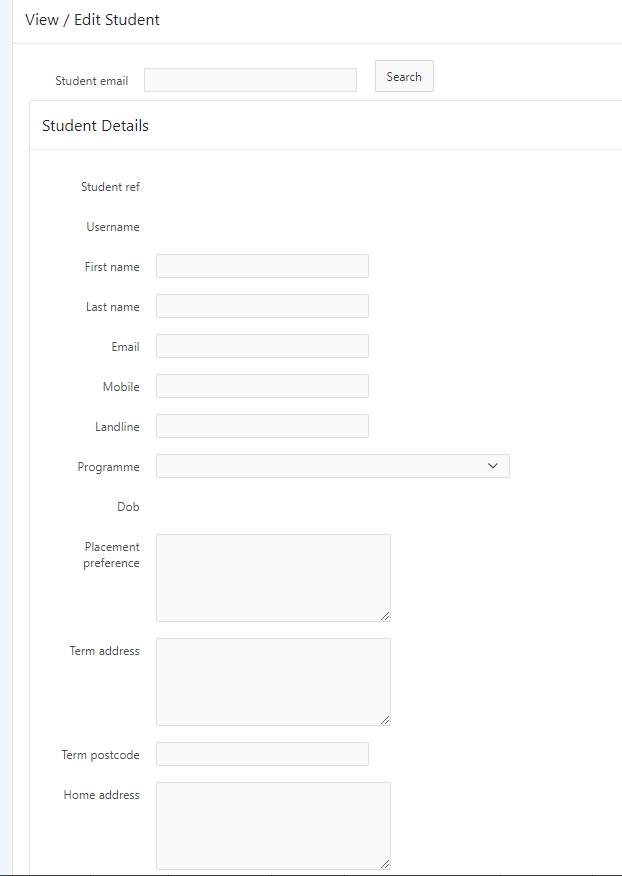
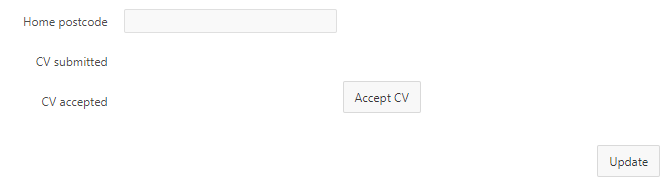


‘View/Edit My Details’ displays the user’s details; this is done by using similar methods for ‘View Job’ page.



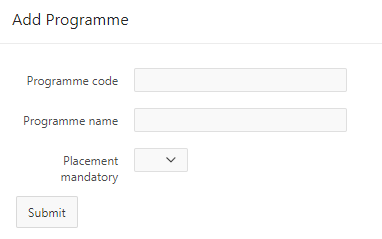
## Admin App Development Screenshots



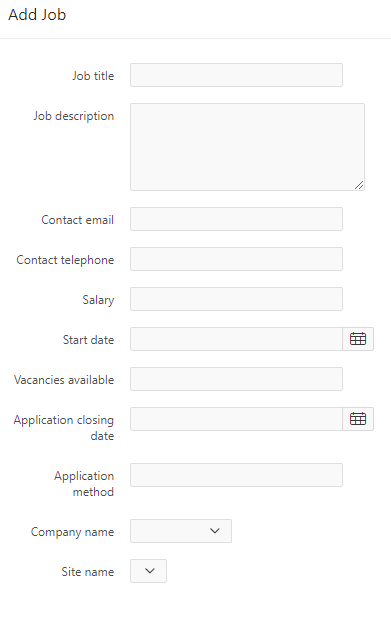
1 – Home page

2 – View/Edit students

Specific pieces of data, such as student reference number and Date of Birth, are uneditable. This is to provide client side data integrity of vital information. When clicking the Accept CV Button, the current date is added as the CV accepted date if there is a CV submitted date.

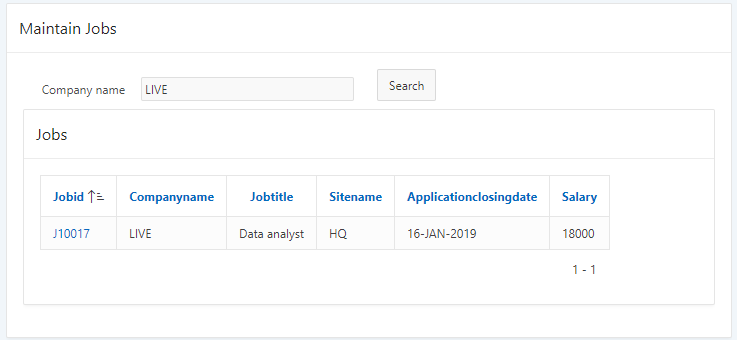
3 – Add programmes

Enables admin to add a new programme to the system.



4 – Add job

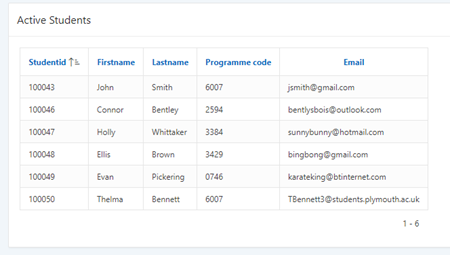
Enables admin to add a job to the system.

5 – Manage jobs

Clicking on a job ID will take the user to another page where they are able to manage the jobs details.

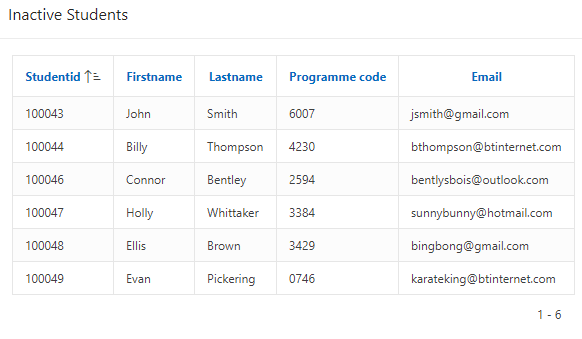
6 – View/edit company

To add a site, the user must first search for a company. The company searched for is then added to the site as a foreign key.

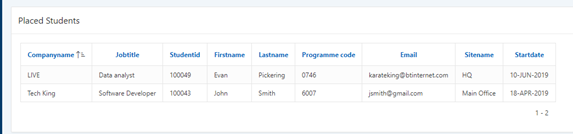


8.1 – Management reports: Active students

Shows a list of active students

8.2 – Management reports: Inactive students

Shows a list of inactive students.



8.3 – Management reports: Placed students

Shows a list of placed students with the details of the job they are placed on.



8.4 – Management reports: Active students not placed

Shows a list of active students that are not placed.

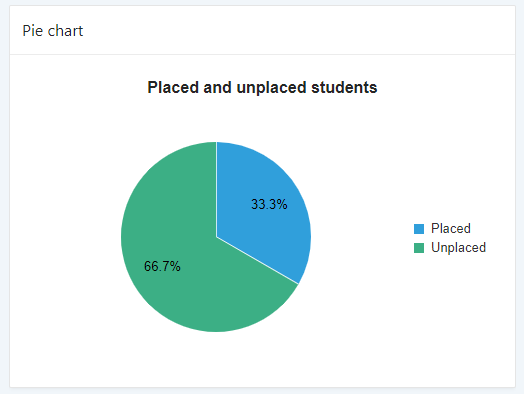
8.5 – Management reports: Closing vacancies

Show a list of job vacancies that are closing within the next 7 days.



9 – Job vacancy calendar

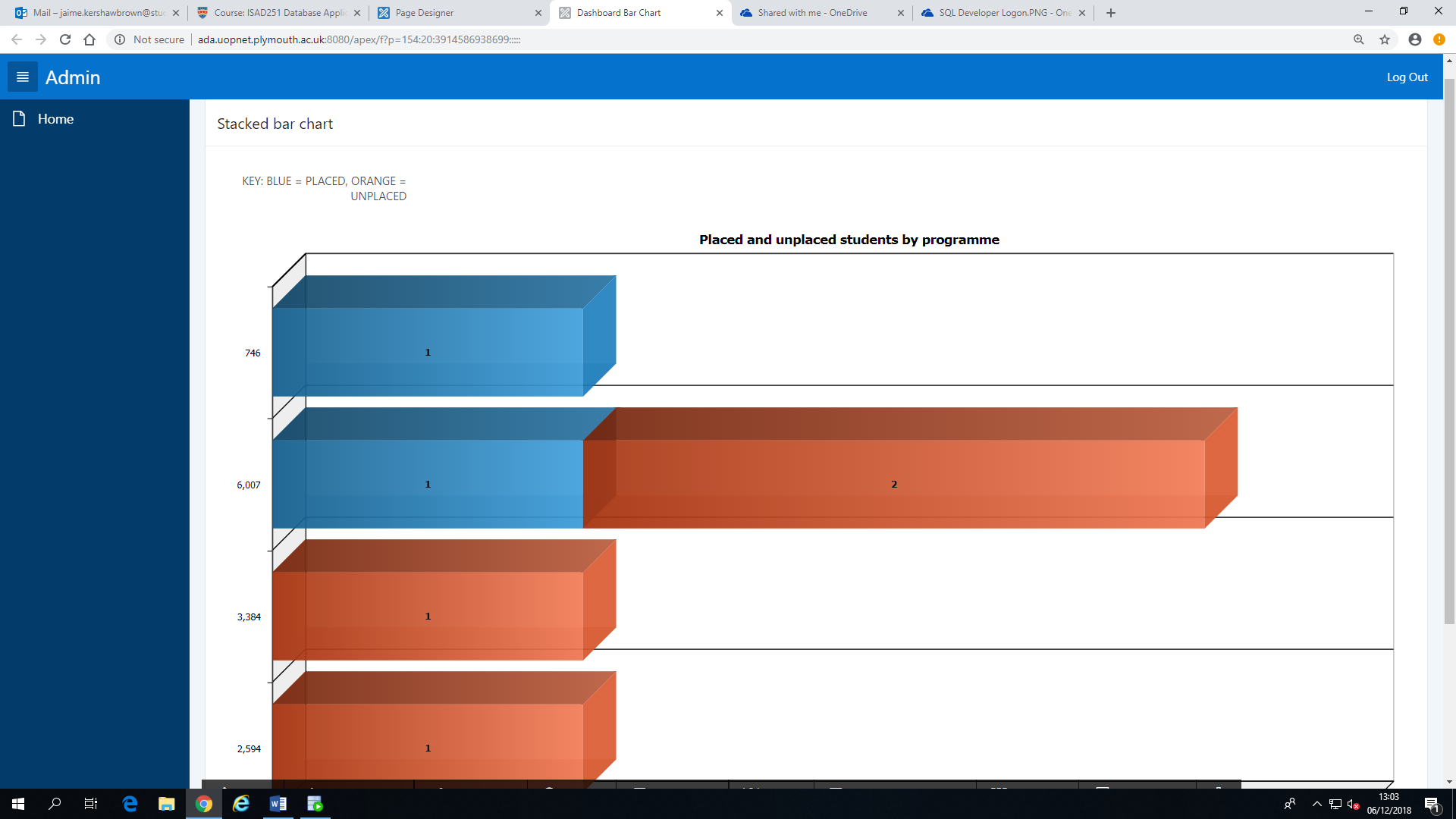
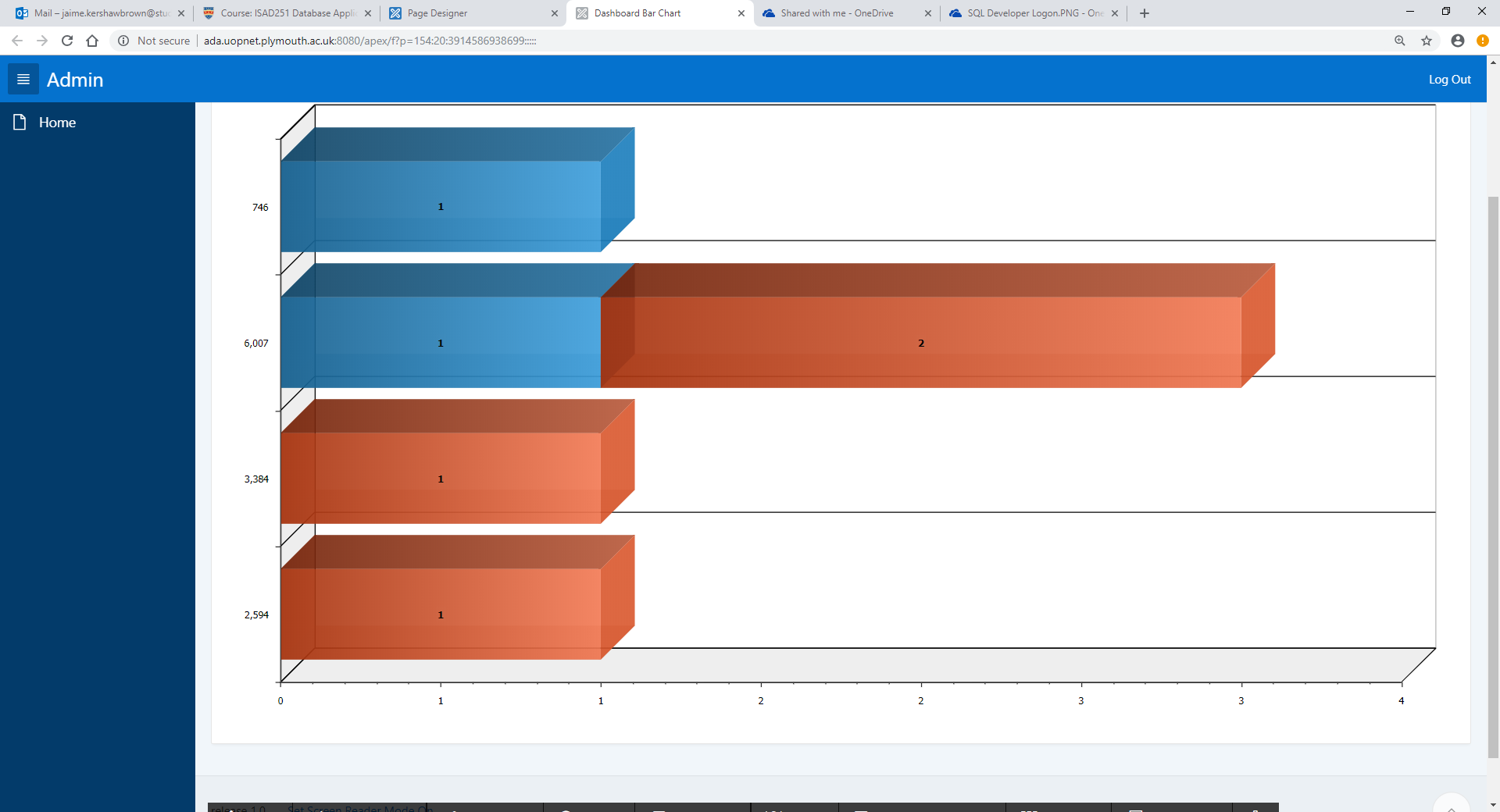
Shows the closing dates of the job vacancies on a calendar with the ability to click on a job to take you to the job vacancy details.

10 – Pie chart

Shows the proportion of place to unplaced active students.

11 – Stacked bar chart

A stacked bar chart to show the ratio of placed students in blue to unplaced students in orange, grouped by programme code.



# Critical Appraisal

In this project, we have created an Oracle SQL Database for a Placement Application Tracking System. Furthermore, we have created both a Desktop Application that will enable the admin to carry out a number of tasks that include: adding and maintain students, jobs and applications; and a Mobile application that will enable students to carry out tasks such as: view and apply to jobs, view and update their application progress. However, this system does have a number of minor issues that could be improved upon in the future.

Firstly, this system has the ability to both add and update students from both the admin and the student application. This means that students and admins will be able to edit a student’s information where the admin accounts have the extra ability to edit for example a student’s name or date of birth should it of been entered incorrectly. Additionally, data integrity has been used where possible which is seen in the programme code section that provides the admin with a drop-down box to enable them to select which programme that student is on so the admin is not required to remember each individual programme code.

Another particularly successful element was the management reports which enable the admin to view tables of students that are active, inactive, placed, and active but not yet placed. Additional information about the student has been displayed such as their email should the admin want or need to contact them. Furthermore, in the management report, there is a closing vacancies section that shows a list of jobs that are closing within the next 7 days which would be useful as the admin could use this information to send out reminders to students.

Another worthy part of the system is the management dashboard. The dashboard includes a stacked bar chart that shows the number of placed and unplaced students of each programme; a pie chart that shows the percentage of the proportion of placed to unplaced active students and a calendar that shows the dates of when each job vacancy closes. Furthermore, on the dashboard calendar, the admin is able to click on each job which will produce a page showing that jobs details.

On the other hand, there are a number of pitfalls with this system. Firstly, the student web application does not implement student login accounts and currently, students are only able to access their accounts by clicking on their name in a drop-down menu which is not sustainable as it makes the system unprotected due to the fact that any student can access another student’s information. Additionally, passwords are also held in plain text inside the database and so makes this system very vulnerable to possible attacks. Therefore it is important that passwords should be encrypted when stored in the database.

Another weakness of the system is that the system only for example records the date on which a job application was submitted or updated and it would be more useful for the system to also record the time of which it was submitted.

Overall, it is clear that this system does meet the criteria set out for it in terms of providing all the basic functionality of maintaining the data within the database. Furthermore, it also provides management reports and a dashboard to enable the admin to view the data in a number of different ways. The key area, however, that does need to be rectified is the ability for students to log in with their specific username and password.