**EITS REPORT**

**Date – 18/06/18**

**AUTHORS**

**JAKOB | BRAYDEN | ALEX**

**Index**

Contents

[**1.0 Preliminary Client Meeting Plan** 3](#_Toc517767622)

[**2.0 Preliminary Analysis Report** 4](#_Toc517767623)

[**3.0 Software Project Management Tools** 16](#_Toc517767624)

[**4.0 Report Methodology** 16](#_Toc517767625)

[**5.0 Project Plan** 18](#_Toc517767626)

[**6.0 Source Code Control and Collaboration Software** 20](#_Toc517767627)

[**7.0 Project Results** 26](#_Toc517767628)

[**8.0 Review of Management Tools** 26](#_Toc517767629)

[**9.0 Technical Report** 27](#_Toc517767630)

[**10.0 References** 59](#_Toc517767631)

# **1.0 Preliminary Client Meeting Plan**

**Hardware - Jakob:** A Desktop computer with specifications to develop the application

**Software - Jakob:** Sourcetree, Microsoft Project, MySQL, and Discord

**Peoples roles - Alex:** Clients, Admins, Caseworkers

**Processes - Brayden:**

* To allow clients, admins and caseworkers to login as their respective role. Admins and caseworkers can see clients log in time.
* Show clients the courses and units for them to enrol in which will be displayed for admins and caseworkers to see.
* Allow admins to import, update and delete client and caseworker data and also industry, courses, and units.
* Allow caseworkers to view client data

**Data:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Users (Jakob)  userID  courses\_id  units\_id  fName  lName  email  password  loginDate  results  notes | Admins (Jakob)  adminID  fName  lName  email  password | Caseworkers: (Brayden)  caseworkerID  fName  lName  email  password | Industries: (Brayden)  Id  industryDesc | Courses  (Alex)  Id  Industries\_id  Code  title | Units (Alex)  Id  Name  unitDesc  courses\_id |

Table 1: Database tables and contents

**Product purpose:**

**Jakob**

The product is designed to assist EITS with client administration and supporting client activity. The users of this application will select from a list of desired industries, within the industries it provides information about educational courses which they can take and the skills required for them.

**Brayden**

The administrator can then view and track this information from the system which recorded the users' name, email, number, and activity throughout the application. EITS would like this application to welcome clients to the office and track their attendance and training progress throughout their use on this application and would like the information to be accessible via her computer tablet to easily access and see the data.

**Information gathering techniques: - Jakob**

For the best and more appropriate information gathering we will conduct a client meeting to firstly gain the information and knowledge necessary to begin construction of the application. We will also use researching to the best of our abilities to gain further information about how the project should be constructed and detailed. Our team has also been given existing repositories of information in the form of a client info and sample solution detail document PDF and a Client sample forms word document which will provide a lot of informative details about what we have to do.

**Questions:**

**Jakob**

* Where does the process start?
* What does this feature need to do?
* How might we meet this business need?

**Brayden**

* Where would the user access this feature?
* Where would the user be located physically when using this feature?
* When will this feature be used?

**Alex**

* Do you have any project documentation you can provide us?
* How many industries did you want us to list?
* How many categories did you want us to list?

**Existing Repositories of information: - Jakob**

EITS Client info and sample solution detail document.pdf

EITS Client Sample Forms.docx

# **2.0 Preliminary Analysis Report**

**Introduction:**

Brayden

Our team of four has been assigned with developing an application designed to assist EITS with client administration and supporting client activity. The users of this application will select from a list of desired industries, within the industries it provides information about educational courses which they can take and the skills required for them.

Jakob

The administrator can then view and track this information from the system which recorded the users' name, email, number, and activity throughout the application. EITS would like this application to welcome clients to the office and track their attendance and training progress throughout their use on this application and would like the information to be accessible via her computer tablet to easily access and see the data.

Alex

Our team must look for the necessary hardware and software like a desktop based database application and interface components to be developed for user access which might be from a computer tablet, laptop or PC and for software being a web-based database and windows based software.

**System Request Summary: - Jakob**

EITS would like a new easily accessed system to track and view attendance, records, and results of the users. This system will need to have a database and GUI built using MySQL and NetBeans to allow EITS to easily view and obtain the necessary information they need.

All requested features include:

* Welcoming clients to the office and tracking their attendance throughout the app
* Interacting with clients about training options based on the selected industry they desire to look at.
* Keeping a record of the clients' choices training and progress through each visit.
* For the system to store record details for later review, reporting and statistical reporting purposes and for later use to plan future company’s shops work practices.

**Findings:**

**Business Requirements: - Alex**

* Who was it built for ?: This application was built for EITS
* Goals: To create a new GUI application to allow clients to view and enrol in courses, admins to edit client and caseworker data and to implement and edit courses, and for caseworkers to view client data, attendance and progress.

**Process Requirements:**

**Brayden**

The application must be able to allow clients, admins and caseworkers to interact with the GUI in an easy manner while also being connected to the database which in some cases they can modify and interact with it. This will be done through the sign up, client data (Admin), course list (Admin), Unit list (Admin) and Industry List (Admin) pages.

**Feasibility:**

**Jakob**

For the project to take place, it must first be deemed technically feasible to develop and economically justifiable. The project must first be checked for not having too many resources before development to ensure the project is worth the investment. We must first check the required hardware, software and technologies to develop this project.

NetBeans, Sourcetree, MySQL, MS Project, and Windows 7,8, or 10 which are all software which is either free to use or the team already has at their disposal so the software and time management tools to develop this project is feasible.

The Hardware to develop this project requires an average system needing no more than 4 Core 2.5Ghz CPU, 4GB of RAM, 100 GB of storage, and a good connection to the internet. The Team already has systems with this level of hardware and more so the hardware to develop this project is feasible.

With the project only requiring a small database and GUI Interface to be developed and connected to each other, it is definitely feasible to construct on time and within budget to what is appropriate for the Project manager.

**Jakob –**

Some Constraints for the project development could be that a member of the development team has left so some features of the application might be delayed somewhat since the other developers must now take on the work of the team member who has left. This will impact the overall time and cost to develop the application.

**Jakob**

**Number of pages:**

* Login
* Sign Up
* Course list
* Course list (Admin)
* Industry List (Admin)
* Unit List
* Unit List Alternate
* Unit List (Admin)
* Admin Panel (Admin)
* Caseworker Panel (Caseworker)
* Client Data (Admin)
* Client Data (Caseworker)
* Client Progress (Caseworker)
* Client Attendance (Caseworker)
* 2 Thank you and 1 error message pages

**Recommendations:**

**Jakob**

Recommendations for going about developing and planning the application is to firstly have easy and effective communication between team members. This would be done through an online chat service, primarily Discord. Discord is a free online chat service which allows multiple people to join in on a call and easily share their screens. This will allow all team members to easily work together and help each other with problems which might arise. Finding and using a quality chart service will improve communication and overall performance of the group leading to a higher quality product.

**Brayden**

Another recommendation is that we must be reasonable about what we can accomplish and not try to produce or develop something that just isn't practical. This will just waste time and resources and lead to a failed project. Instead, we must develop an application which is within our time and cost budget and can be done by all team members.

**Time and Cost Estimates:**

Brayden

For our team to complete this project on time and within the budget we will have planned and utilised the necessary tools, programs, hardware and tactics. This will also give us a good understanding of the total cost and budget for the project and how long the development will take and assist us from running late and into delays. To do this we would have used programs such as Microsoft Project to plan out our schedules and deadlines for each task and to allocate costs to each task to get an accurate budget which the team must remain in. As a development team our primary goal is to make profits, and doing so requires restricting ourselves and having a clear idea for everything which the client wants developed to find the overall cost for the project.

**Alex**

Using the project management tool Microsoft Project we have created an estimate for the time and cost it will take to complete our project. We took in to account the required work requested by the client such as the number of pages and features needed and also the implementation of the database, our limitations as a group also including a team member which has left, and the required hours it would take for each developer to get the application developed in time.

**Jakob**

Our total cost and time estimate for the project is $17,200. This is estimated from the time frame of the 18th of May to the 18th of June or 31 days at an hourly rate of $100 while the initial client meeting is $50 an hour.

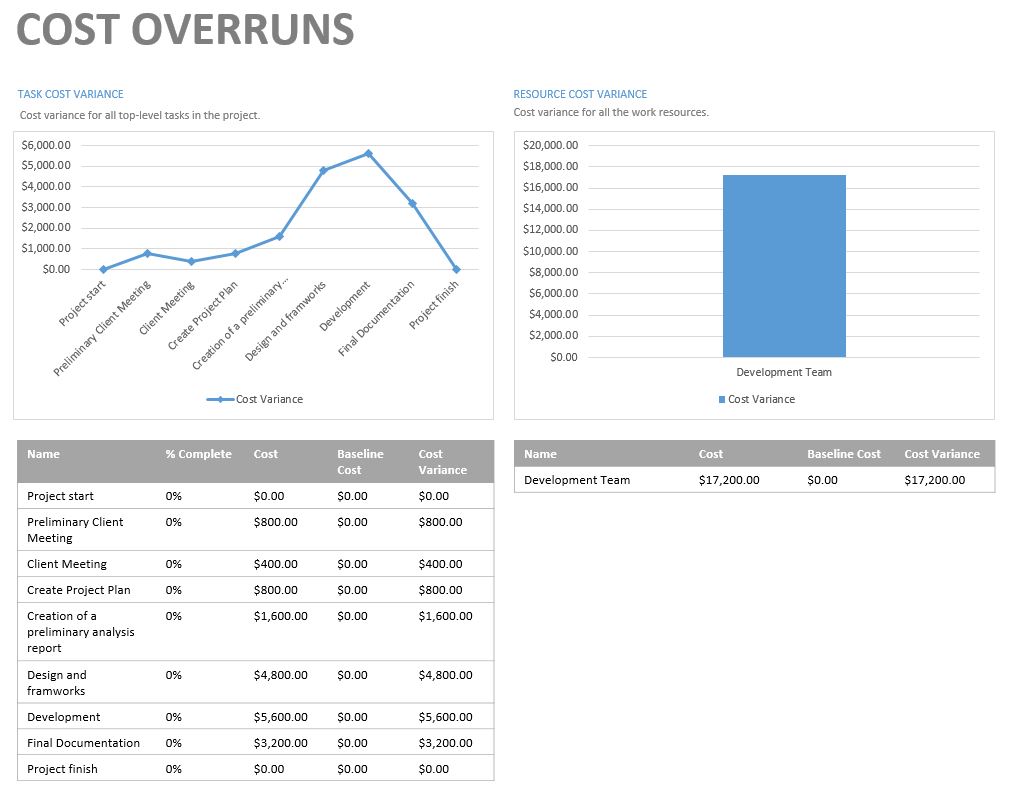


Figure 1: Cost Overruns and total predicted Cost of project

**Expected Benefits:**

**Alex**

The expected benefits for creating this application and database is that it will boost student enrolments and overall sales by improving access to information and allowing them to easily navigate and discover information about the industries, courses and their units, and if it will be a good fit for them overall increasing profits for EITS.

With the creation and linking of the database it will also increase organisational productivity and overall organisation since now all EITS admins and caseworkers can easily navigate to and see client enrolments and client data.

**ER Diagram (Or Equivalent)**

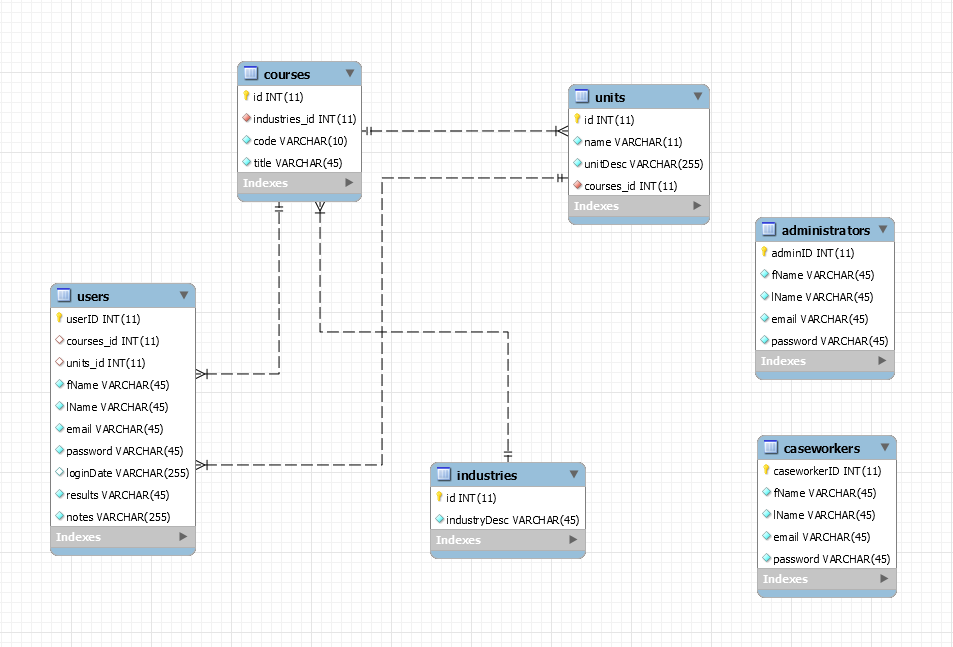
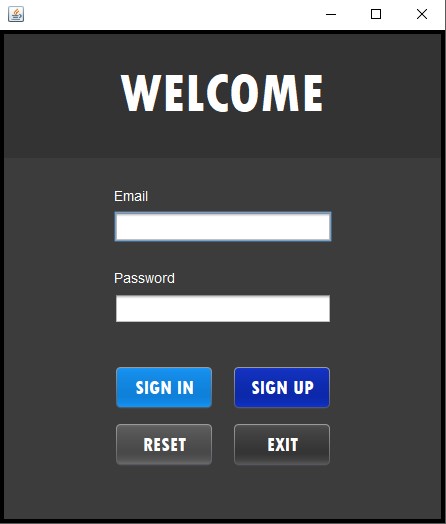


Figure 2: Completed ER Diagram

**Brayden**

**Screen Designs:**

**Jakob**

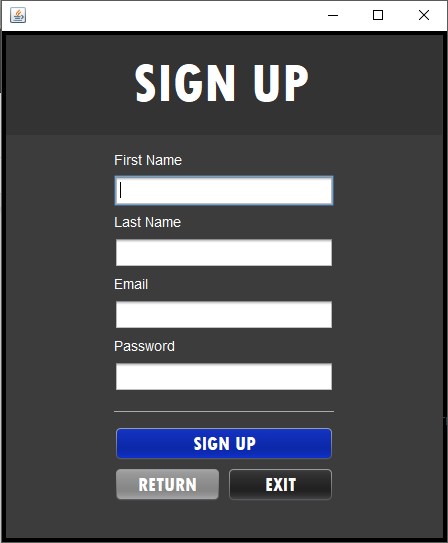
**Design:** The specific colours and design of each page have been selected to create and allow for a simple and easy to use experience for clients, caseworkers and admins. This has been achieved by implementing colours which pare together nicely and a dark theme to protect against eyestrain for users and to make it easy on the eyes. Each page remains consistent with its layout and colours to keep a flowing and familiar feel to the application.

**Login Screen - Jakob**

Functionality: The functionality for this page is to link and tie all pages together. The welcome page is used to enter an email and password which has already been given to you inside the database. The welcome screen displays four buttons

* Sign in: Used to log the user in from the database
* Sign up: Used to put a user in to the database.
* Reset: Used to reset the text inputted
* Exit: Used to exit the application.

Figure 3: Login or Main menu Page



**Sign Up Screen - Jakob**

Functionality: The functionality for the signup page is to allow a user to input their first name, last name, email and password and for it to be registered inside the database to be later used to login with. The Signup screen displays three buttons

* Sign up: Logs the information given in to the database
* Return: Takes the user back to the Login screen
* Exit: Used to exit the application

Figure 4: Sign Up page to register and log new users

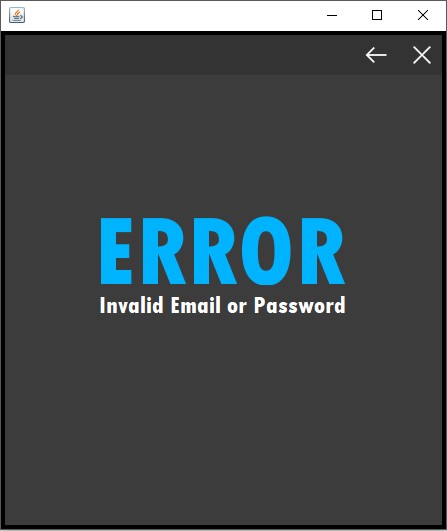
**Error Invalid Email or Password** **- Alex**

Figure 5: Error page to inform user of invalid email or password

Functionality: This page is used to alert the user that their email or password which they entered in to the login page is incorrect or invalid. The error page displays two buttons

* Return: Takes the user back to the Login screen
* Exit: Used to exit the application

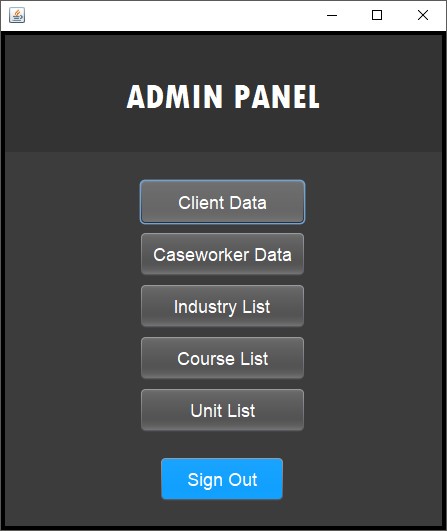
**Admin Main page - Alex**

Figure 6: Admin Main page

Functionality: This page is the main panel display for admins after they login. They can see all panels and controls they have access to and can also sign out. There is 6 buttons on this page.

* Client Data, Caseworker Data, Industry List, Course List, Unit List take you to their respective pages for the admin to edit and view the information.
* Sign out: Signs out the admin and takes you back to the login page

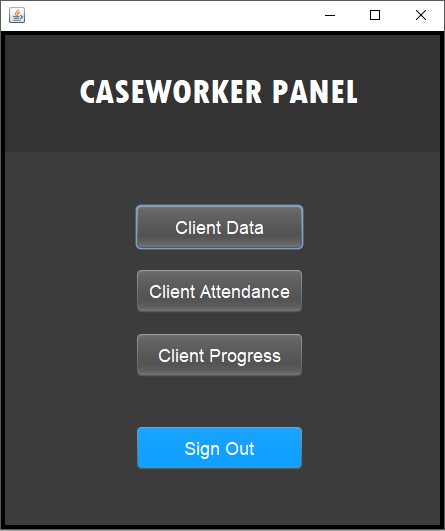
**Caseworker Main page - Alex**

Figure 7: Caseworker main page

Functionality: This page is the main panel display for caseworkers after they have successfully logged in. They can all the panels and controls they have access to and can also sign out. Unlike admins caseworkers cannot edit, insert, or update data but only view it. There is 4 buttons on this page.

* Client Data, Client Attendance, Industry List, Course List, Unit List take you to their respective pages for the admin to edit and view the information.
* Sign out: Signs out the admin and takes you back to the login page

**Client Data Admin page - Brayden**

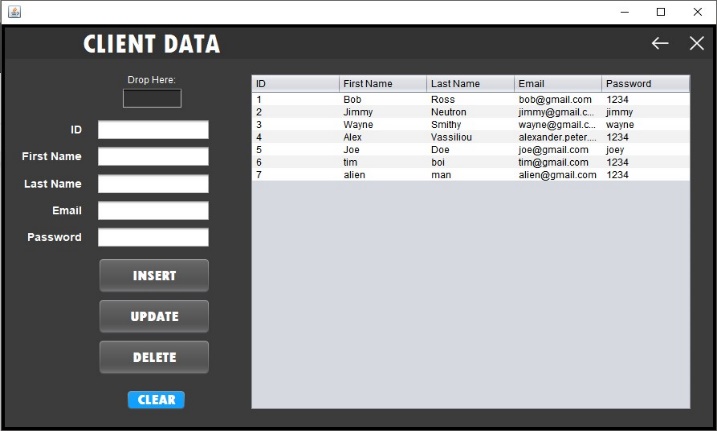
Functionality: This page allows an admin to view, insert, update or delete client data inside the database. The Client data admin page displays 6 buttons and a drop here box.

Figure 8: Client Data Admin Page

* Drop here: Drag and drop information from the table in to the drop here box to display the information in the boxes underneath
* Clear: Clears information inside the text boxes
* Insert: Inputs text inside the text boxes in to the database
* Update: Updates the existing information with modified text inside the text boxes.
* Delete: Deletes the selected client data from the database
* Back: Takes you back to the admin main GUI
* Exit: Closes the application

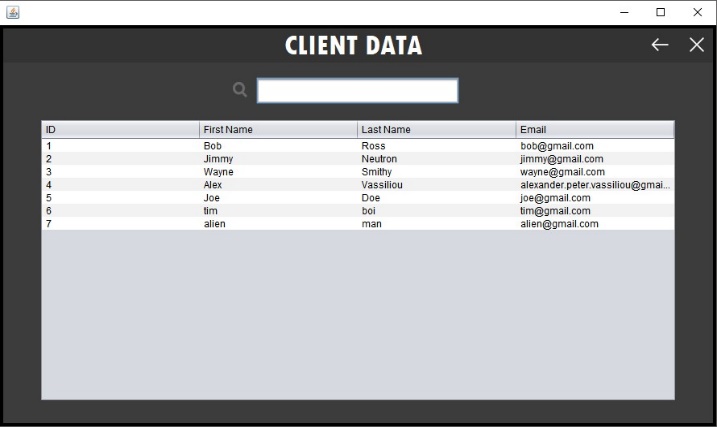
**Client Data Caseworker Page - Brayden**

Figure 9: Client Data Caseworker page

Functionality: This page allows a caseworker to only view and search the client data but does not allow them to insert, update or delete any information. The client data caseworker page displays 2 buttons and a search box

* Back: Takes you back to the caseworker main GUI
* Exit: Closes the application
* Search: allows you to type in a client’s name and it automatically appears at the top.

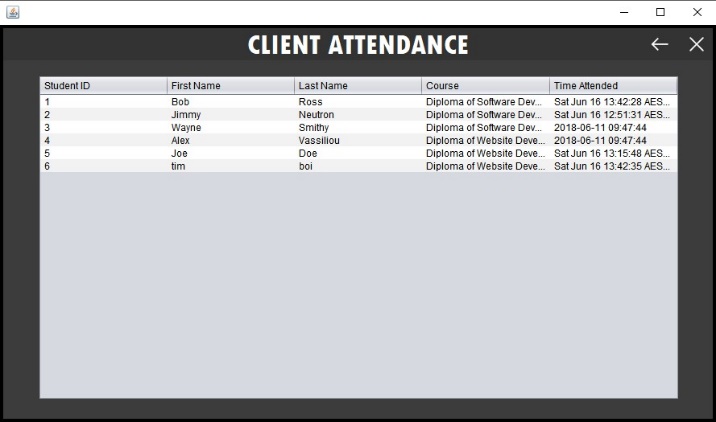


Figure 10: Client Attendance Page

**Client Attendance Page - Alex**

Functionality: This page allows a caseworker to view and search individual client attendance throughout the application and see when clients login. This page has 2 buttons

* Back: Takes you back to caseworker main GUI
* Exit: Closes the application

  
**Client Progress Page - Alex**

Figure 11: Client progress page

Functionality: This page allows a caseworker to view and search individual client progress, enrolments and results. This page displays 2 buttons

* Back: Takes you back to caseworker main GUI
* Exit: Closes the application

  
**Client Course List Page - Jakob**

Functionality: This page is the main page after the login screen clients will see when signing in. A client is displayed two courses to choose from. This page displays 4 buttons

* Sign Out: Takes you back to login page
* Exit: Closes the application
* First and second course: Takes you to the first and second course to choose what you want to enrol in.

Figure 12: Client course list page

**Client Unit List Page - Jakob**

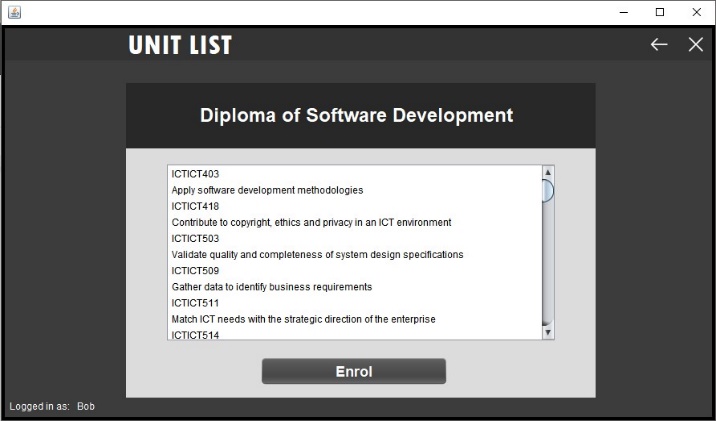
Functionality: This page is the unit list page which displays all the units within the selected course. From here the client can decide to enrol in the course and it will be updated in the database. There is also a client unit list alternate page which is for the second course but is the same layout as this page. There is 3 buttons

Figure 13: Client unit list page

* Back: Takes you back to client course list page
* Exit: Closes the application
* Enrol: Enrols the logged in user in the certain course they picked.

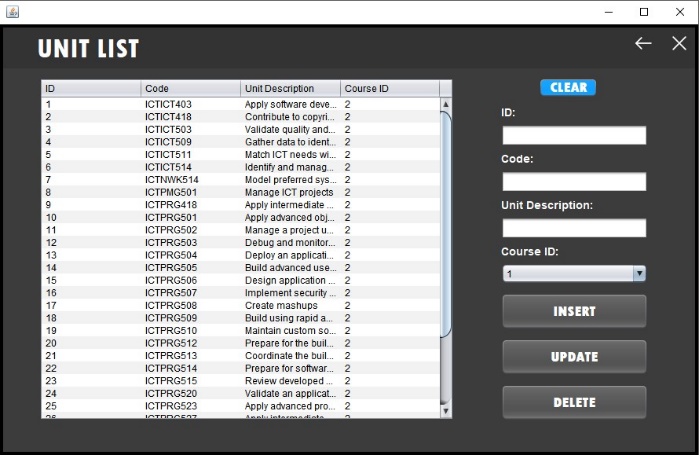
  
**Client Unit List Admin Page - Jakob**

Figure 14: Client unit list admin page

Functionality: This page is the unit list admin page and displays all the units for each course for the admin to insert, update, and delete original or new units. There are 6 buttons on this page

* Back: Takes you back to Admin Main page
* Exit: Closes the application
* Clear: Clears information inside the text boxes
* Insert: Inputs text inside the text boxes in to the database
* Update: Updates the existing information with modified text inside the text boxes.
* Delete: Deletes the selected unit data from the database

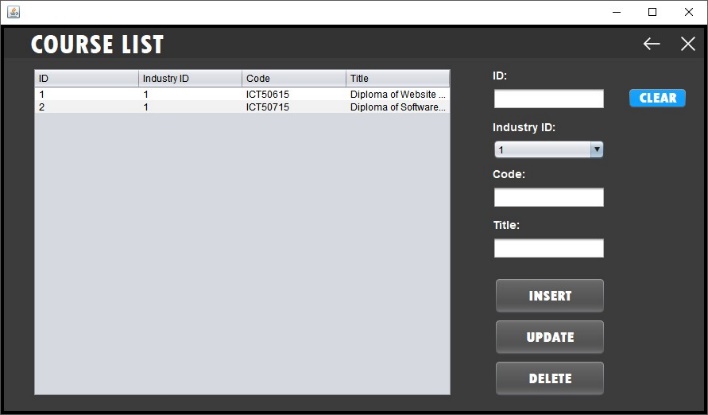
**Client Course List Admin Page - Jakob**

Figure 15: Client Course list admin page

Functionality: This page is the course list admin page and displays all the courses for each industry for the admin to insert, update, and delete original or new courses. There are 6 buttons

* Back: Takes you back to Admin main list page
* Exit: Closes the application
* Clear: Clears information inside the text boxes
* Insert: Inputs text inside the text boxes in to the database
* Update: Updates the existing information with modified text inside the text boxes.
* Delete: Deletes the selected course data from the database

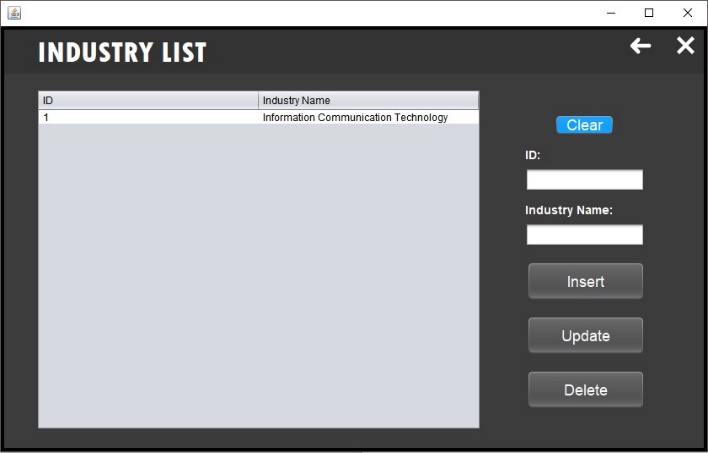
**Client Industry List Admin Page - Brayden**

Figure 16: Client industry list admin page

Functionality: This pages is the industry list admin page which allows admins to insert, update and delete existing or new industries in the database. This page consists of 6 buttons

* Back: Takes you back to admin main page
* Exit: Closes the application
* Clear: Clears information inside the text boxes
* Insert: Inputs text inside the text boxes in to the database
* Update: Updates the existing information with modified text inside the text boxes.
* Delete: Deletes the selected industry data from the database

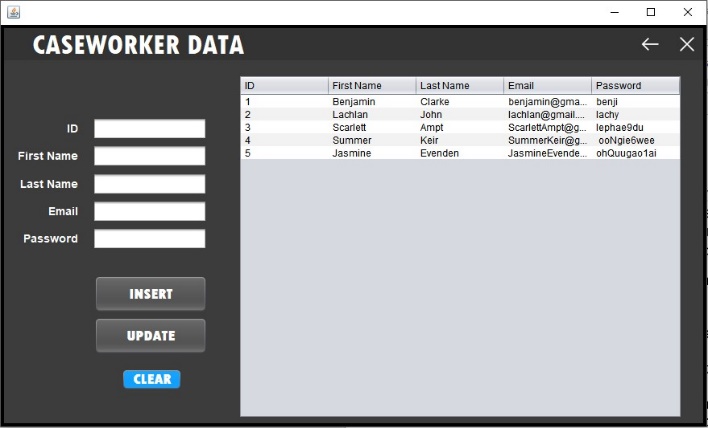
**Caseworker data Admin Page - Brayden**

Figure 17: Caseworker data admin page

Functionality: This page is the caseworker data admin page which allows admins to insert and update existing or new caseworkers in to the database. Admins cannot however delete existing caseworkers from the database. There is 5 buttons on this page

* Back: Takes you back to admin main page
* Exit: Closes the application
* Clear: Clears information inside the text boxes
* Insert: Inputs text inside the text boxes in to the database
* Update: Updates the existing information with modified text inside the text boxes.

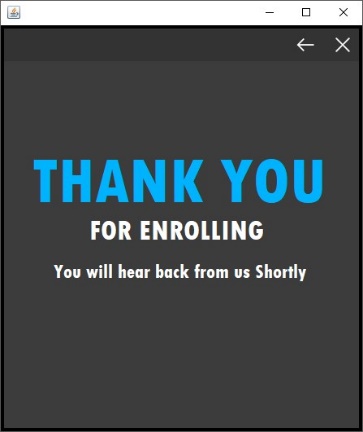


Figure 18: Thank you enrolment page

**Thank you enrolment page – Brayden**

Functionality: This page is the thank you for enrolling page which displays after a client has enrolled in a course. There is two buttons on this page

* Back: Takes you back to course list page
* Exit: Closes the application

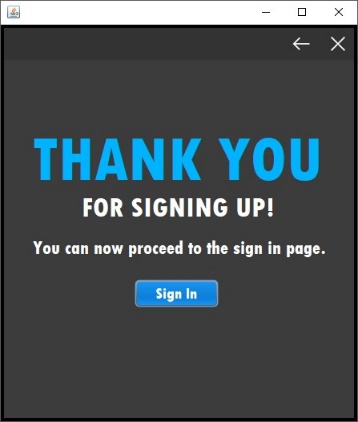
**Thank you sign up page - Brayden**

Figure 19: Thank you sign up page

Functionality: This page is the thank you for signing up page and displays after a client has successfully sign up their account. There is three buttons on this page

* Back: Takes you back to the sign up page
* Exit: Closes the application
* Sign In: Takes you to the login page so the client can sign in with their new account.

**Project Analysis Sign-off Sheet**

|  |  |
| --- | --- |
| **Project Name:** EITS Preliminary Analysis Report | **Project Manager:** David Hunt |
| **Start Date:** 22/05/18 | **Completion date:** 23/05/18 |
| **Project Duration:** 22 Hours | **Version:** 1.0 |
| **Project Goal:** The project goal for the Preliminary Analysis report was to construct and design an analysis report which effectively demonstrated, explained, and outlined our findings, recommendations, time and cost estimates, system request summary, expected benefits preliminary screen designs and rough ER diagram. | |
| **Project manager Name and Signature:**  Name:  Signature: | **Development team name and signature:**  Name:  Signature: |

Table 2: Project Analysis sign off table

# **3.0 Software Project Management Tools**

**Sourcetree - Jakob:** Sourcetree is one of the software project management tools which would be utilized to increase productivity, collaboration management, and source code control. Sourcetree allows for you to easily interact with your Git repositories on projects so you can modify and create new code and easily push it to your team. Sourcetree provides Source code control to allow teams to view and keep track of changes to the code. This allows members to easily check back at old code and compare the differences letting them see the changes which have been made.

**Microsoft project - Alex:** Microsoft is another software project management tool which we would utilize to help organise and focus the team. Microsoft project developed by Microsoft is primarily designed to help the project manager in developing a plan, managing the budget, and attaching people or resources to tasks. We will use Microsoft project to find the estimated cost, time and features for the project we will be undertaking. This will give the whole development team a better understanding of when features have to get done and what to do after that task. This will increase productivity and team management while decreasing confusion throughout development since everyone will have a clear understanding of the deadlines and features which they must do.

**JIRA - Brayden:** JIRA is another great software project management tool which we could use in the development of the project. JIRA is an issue and feature tracking product developed by Atlassian. JIRA provides an easy to use interface which provides bug tracking, issue tracking and requested features/updates to parts of the project. JIRA is great for collaboration management since everyone on the team can view small posts from each other about the current bugs, issues or requests and their priority. Once a team member decides to take on the challenge of fixing an issue someone else has put on to the board it will be marked with them working on that certain issue. Using JIRA allows the team to very easily see what need to be changed, added, who requested it and who will be working on the issue. JIRA will increase productivity by improving communication to give team members instant updates on work which must get done and how important that issue/feature might be.

# **4.0 Report Methodology**

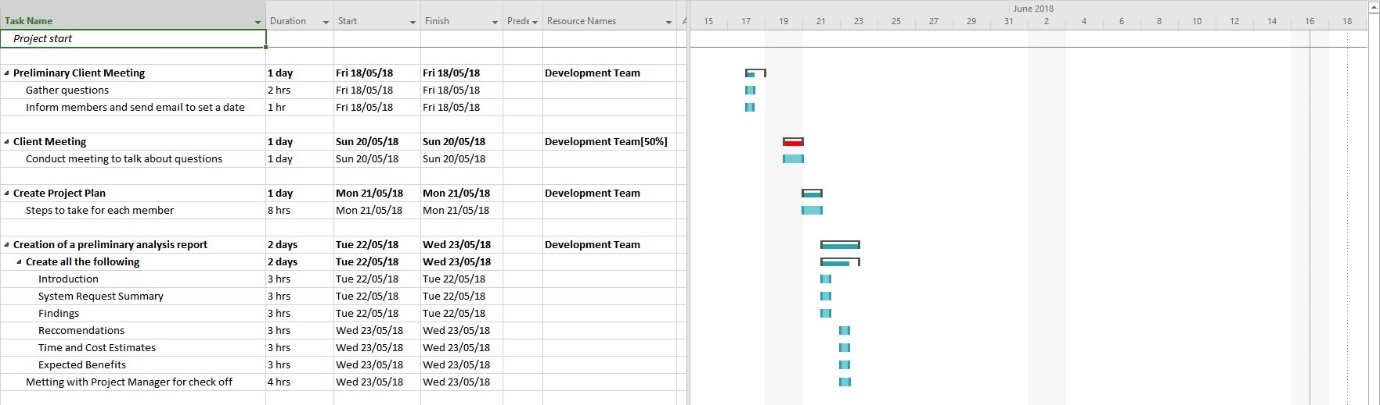
**Jakob**

Methodology we will or would be utilizing for the development of the project consist of gathering information and data through interviews with the client to understand what is necessary and required for the development. Research will also be conducted and we will be searching for similar projects conducted, looking for costs and quality of the product. By doing this it will give our team a better guideline for the industry standard for the task we are taking on, and how to better prepare and plan.

Alex

To avoid source code conflicts we are making sure when committing we are all informed before we have started on the certain commit, there will concurrent updates to our fellow teammates, till the certain commit is finished. Then as a team we also make sure not to be working on the same commit together, commits must be split as if they are both being done at the same time there could be a commit confliction and an unsuccessful merge. Source Code Control is extremely important when it comes to having a successful project and getting your project to work successfully

# **5.0 Project Plan**



**Brayden**

**Alex**

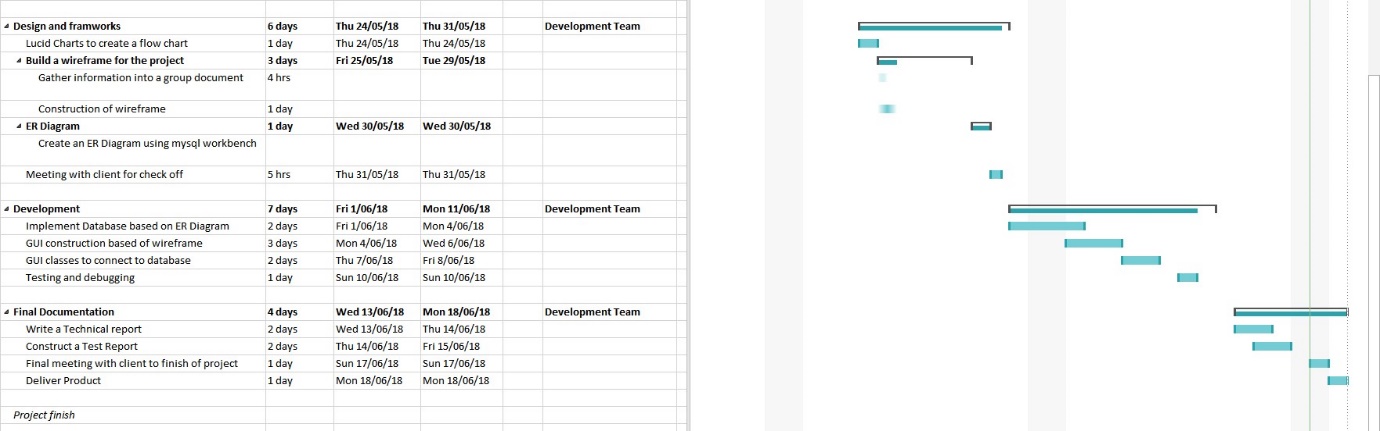


Figure 20: Completed project plan constructed by Alex, Brayden and Jakob



**Jakob**

# **6.0 Source Code Control and Collaboration Software**

Demonstrating use of collaboration software and sharing our screens (Discord)

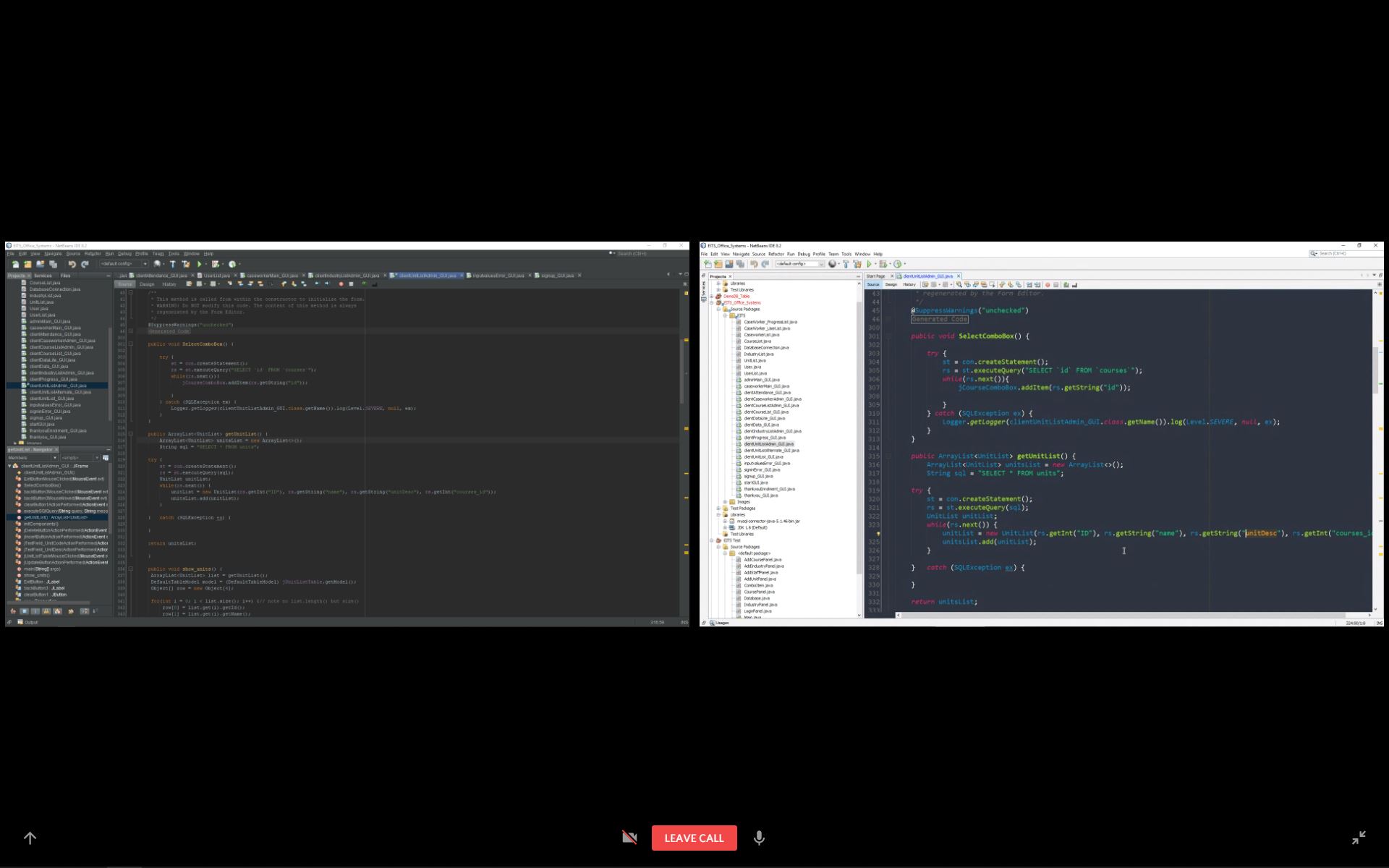


Figure 21: Using Discord (Collaboration software) to develop the application

Demonstrating use of project management and Source code control software using Sourcetree: Jakob

(Video of using Sourcetree and committing) <https://www.youtube.com/watch?v=NdvtumdU57s>

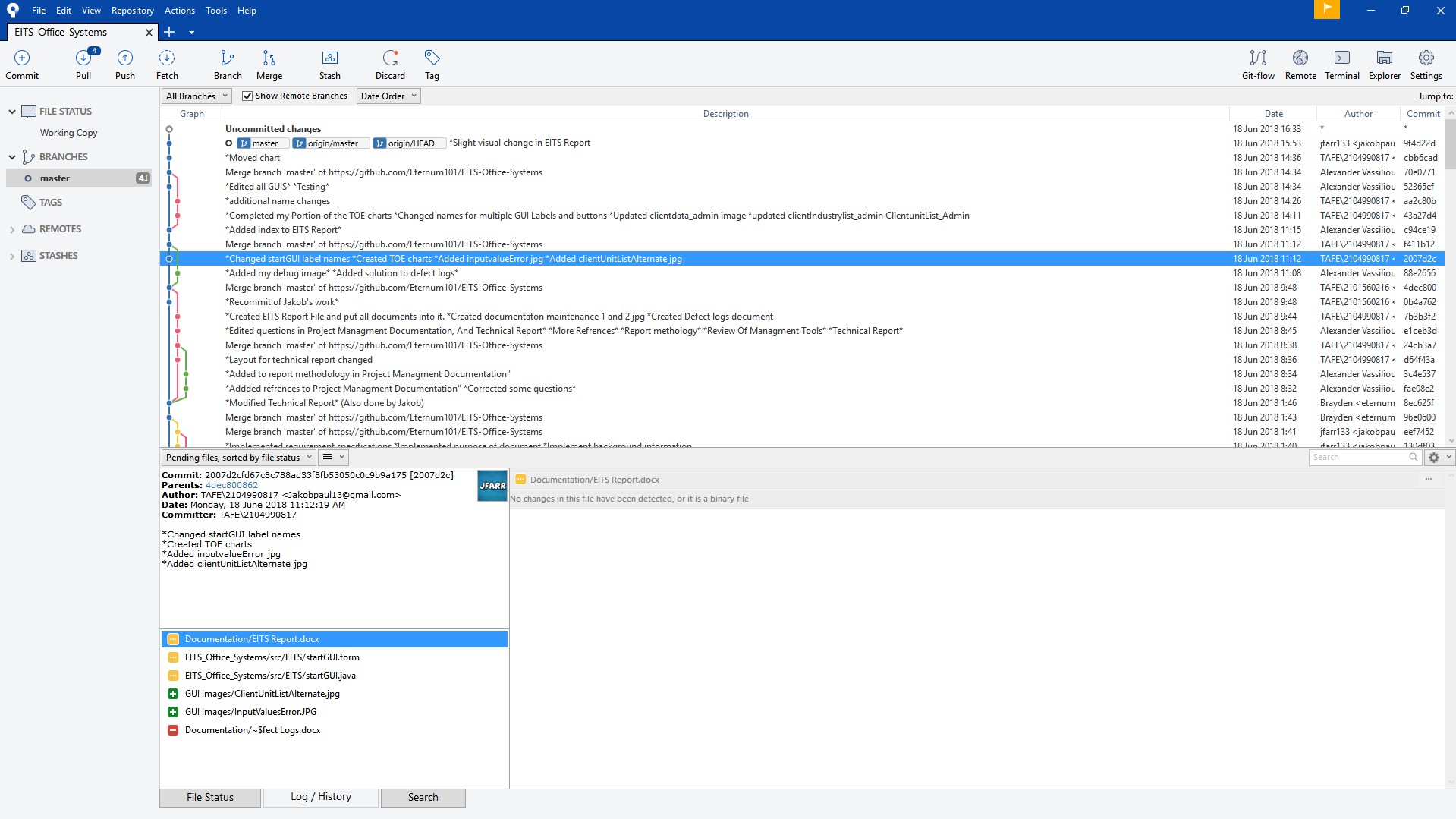


Figure 22: Jakob - Demonstrating use of Sourcetree and having performed commits and pushes

Alex -

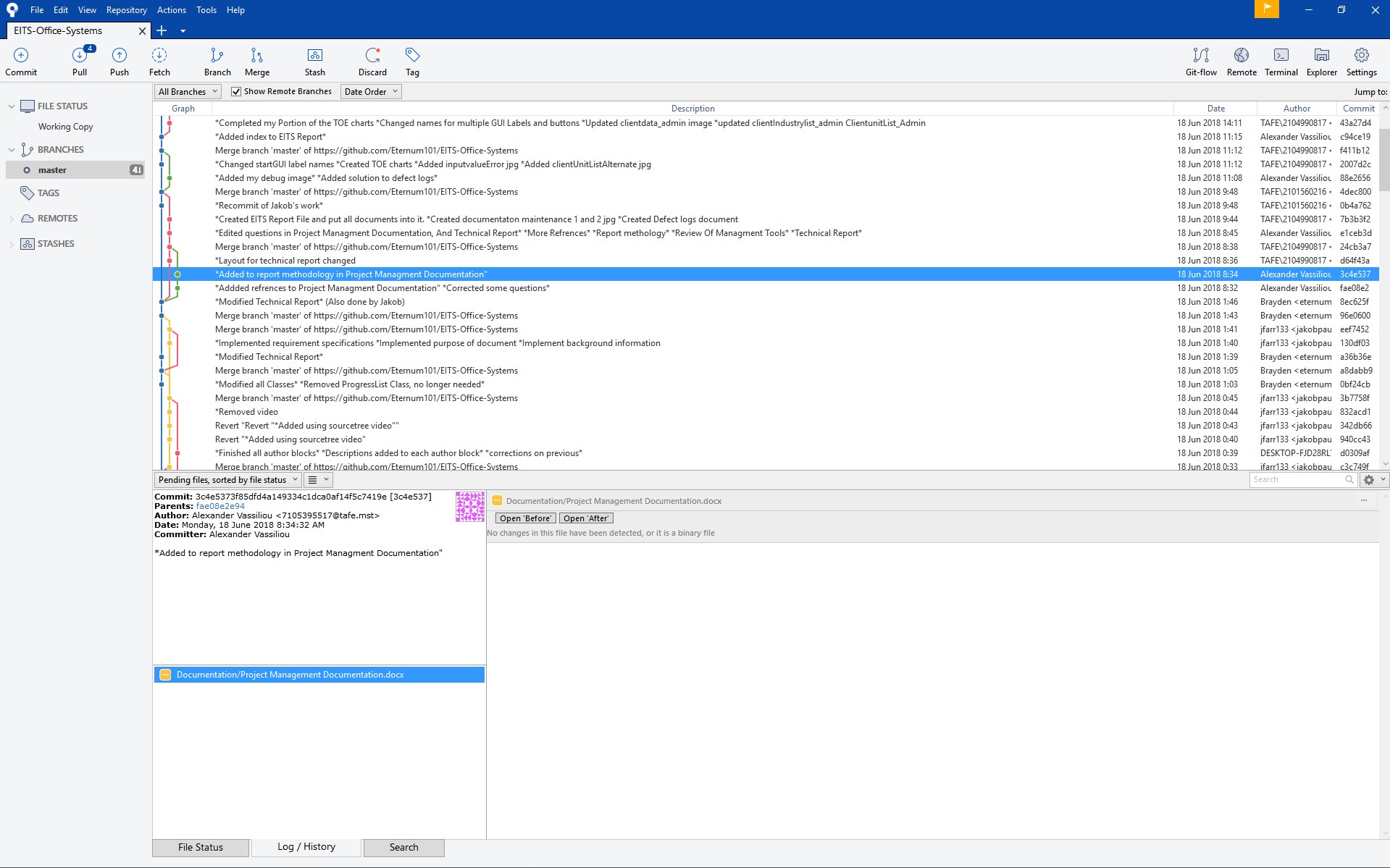


Figure 23: Alex - Demonstrating use of Sourcetree and having performed commits and pushes

**Brayden**

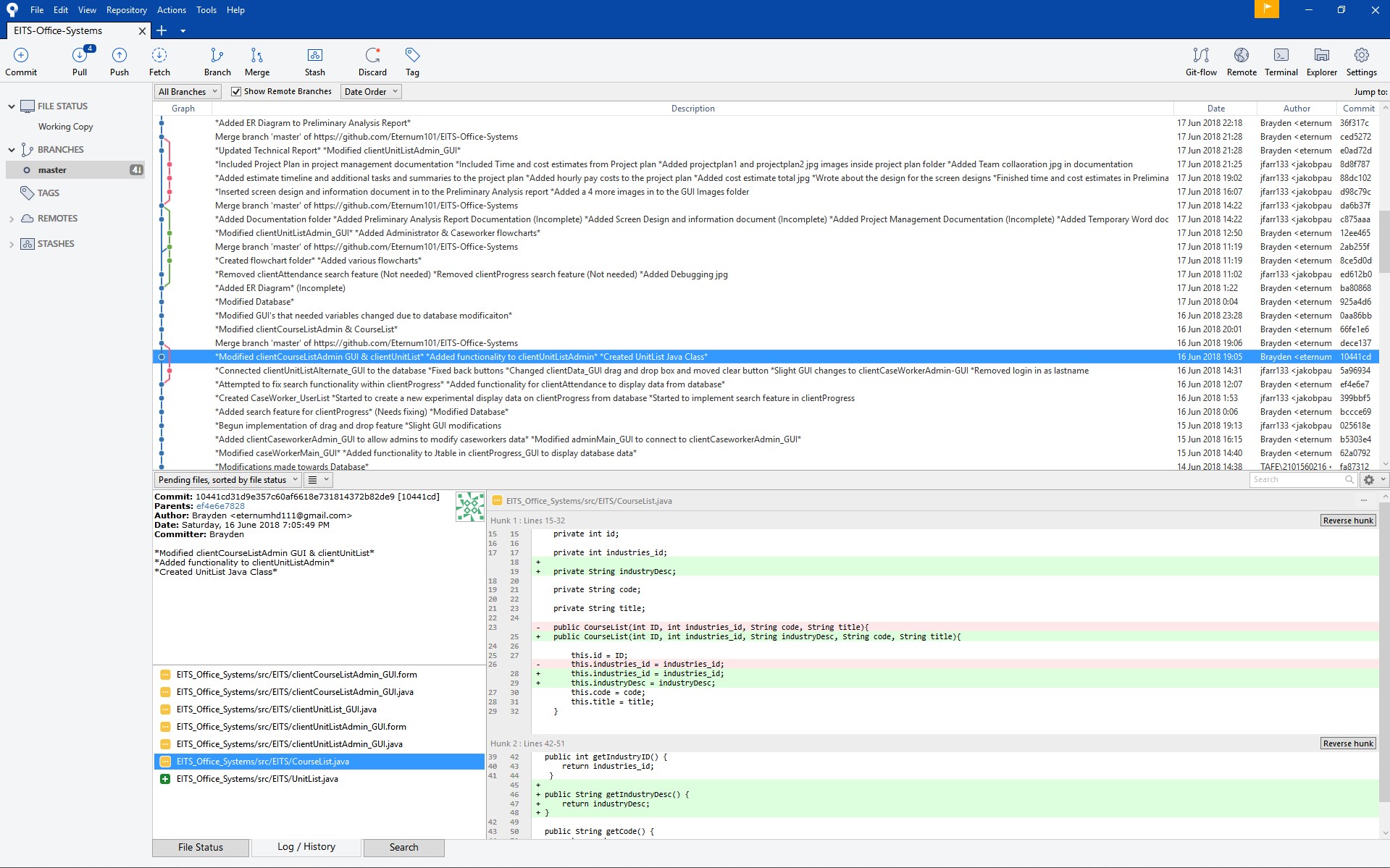
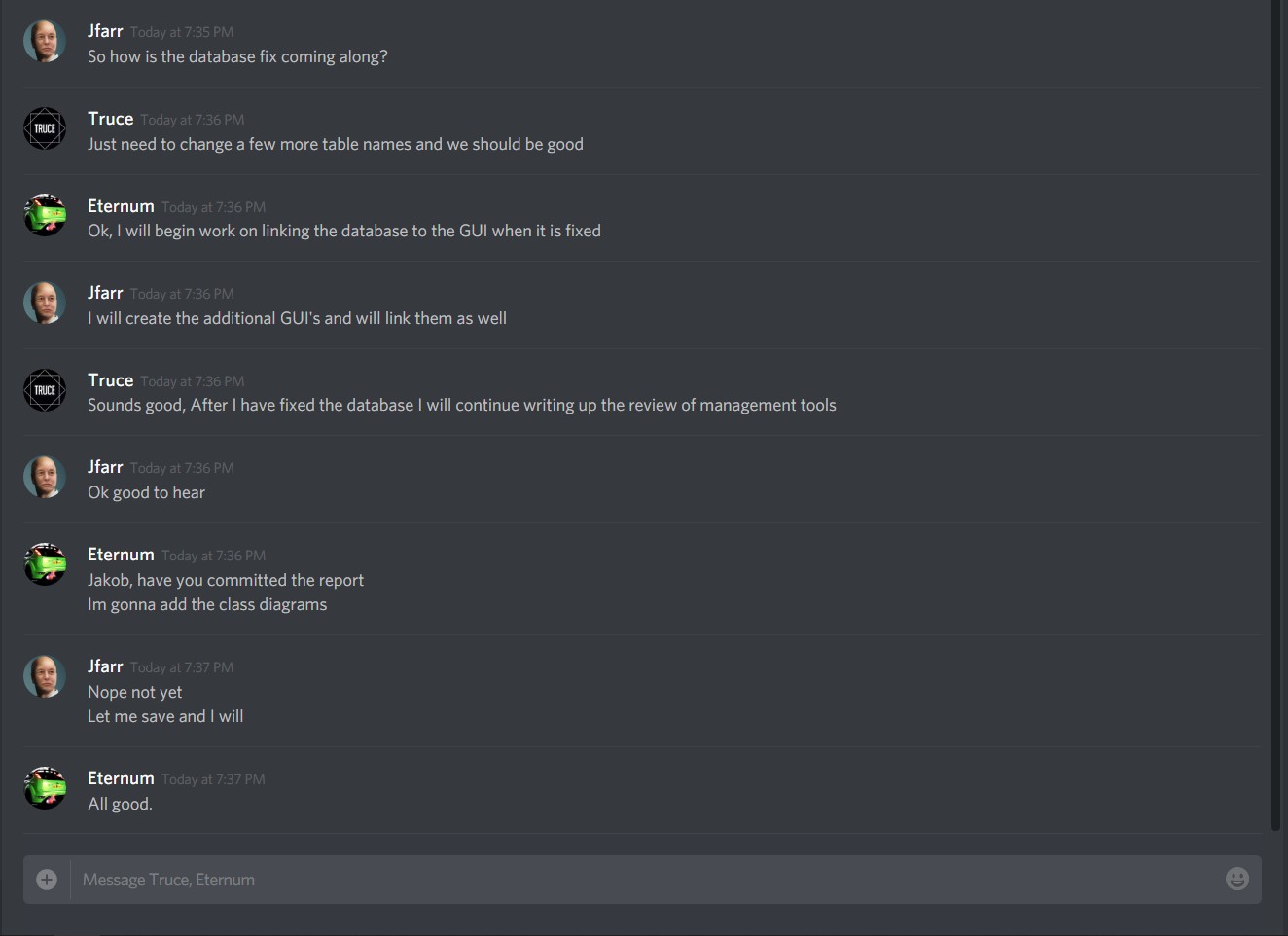


Figure 24: Brayden - Demonstrating use of Sourcetree and having performed commits and pushes

**Demonstrating team communications** 

Jakob

Alex

Brayden

Jakob

Alex

Jakob

Brayden

Jakob

Brayden

Figure 25: Alex, Brayden, Jakob demonstrating use of team communication

# **7.0 Project Results**

**Brayden**

The final project results for our application development has been a successful but challenging task to undertake. The team has found that the initial planning phases which were completed before the development of the project was a big help towards completing the task on time and done so with a high-quality finish. This is not to say there has been no challenges and adjustments made throughout the development of the project. An issue which we had to overcome was getting the login page to work correctly.

**Jakob**

We were having trouble with the error page displaying when the user enters incorrect values or data which was not in the database. We finally overcame this issue by collectively working as a team, utilizing NetBeans Debugging tools and research to fix the issue. Adjustments made to the project would be the time which it took to finish certain features. Some features took longer than expected to complete so adjustments had to be made to the timeline to fit the delays.

# **8.0 Review of Management Tools**

**Jakob**

The management tools used have been effective at increasing productivity, team effort and cooperation. The tools utilized Sourcetree, Microsoft Project and google docs have been used throughout the development of the application and have had their pros and cons. The pros of the management tools used is that with their easy to use UI and features it has allowed for increased productivity for the development of the application and allowing for quality source code control to allow us to view changes and modifications by team members.

**Alex**

We could have also used JIRA, JIRA would have helped in a few way by improving the communication rather than source tree by having similar but more improved updates to team member, and with Jiri it adds a much more dynamic view of progress by fellow team members, by being able to see what someone’s working on as they are working on it rather than having to ask or wait for them to finish to see it update on the screen. Having Jiri, would increase the productivity and communication for our group, and made it only better and more efficient for the client.

# **9.0 Technical Report**

**Purpose of the document: - Alex**

The purpose of the document is to define the current and future business processes.

**Background information - Brayden**

The product is designed to assist EITS with client administration and supporting client activity. The users of this application will select from a list of desired industries, within the industries it provides information about educational courses which they can take and the skills required for them. The administrator can then view and track this information from the system which recorded the users' name, email, number, and activity throughout the application. EITS would like this application to welcome clients to the office and track their attendance and training progress throughout their use on this application and would like the information to be accessible via her computer tablet to easily access and see the data.

**Requirements specifications:**

Project Features:

**Client: - Brayden**

**Login:** The login will be used to sign a client up which has registered in the sign up page

Flow of events:

* Client launches the application
* Login screen is displayed
* Client inputs their credentials and clicks the sign in button
* Client is signed in (Logged in the database) and can assess courses

**Sign Up:** The sign up will be used to register clients to the database with the information they have inputted

Flow of events:

* Login screen is displayed
* Client clicked the sign up button
* Client inputs their data into the text fields
* Client clicks sign up button and the data is sent in to the database
* The client is now registered and can sign in

**Course List:** The course list will be used to display the courses within a specific industry

Flow of events:

* Login screen is displayed
* Client signs in with their existing credentials
* Client is taken to the course list to select a course they would like to take
* Client can also sign out from this page

**Unit List:** The unit list will be used to display the units within the course the client has selected

Flow of events:

* After signing in the client will select a course
* When selecting a course the unit list will be displayed
* The client can then proceed to click enrol in the certain course (This will be logged in the database)

**Admin: - Jakob**

**Login:** The login will be used to sign the admin up which has registered in the database

Flow of events:

* Admin launches the application
* Login screen is displayed
* Admin inputs their credentials and clicks the sign in button
* Admin is signed in (Logged in the database) and can assess the admin panel

**Admin Panel:** The admin panel will be used to display all the options and features the admins have access to.

Flow of events:

* Admin signs in using their credentials
* Admin is brought to the admin panel for them to select between a few buttons.
* Admins can then insert, update and delete client and caseworker data as well as industries, courses, and units.
* Admin can then sign out from this page

**Caseworker: - Alex**

**Login:** The login will be used to sign in the caseworker which has registered in the database

Flow of events:

* Caseworker launches the application
* Login screen is displayed
* Caseworker inputs their credentials and clicks the sign in button
* Caseworker is signed in (Logged in the database) and can assess the Caseworker panel

**Caseworker Panel:** The caseworker panel will be used to display all the options and features the caseworkers have access to.

Flow of events:

* Caseworker signs in using their credentials
* Caseworker is brought to the caseworker panel for them to select between a few buttons.
* Caseworkers can then view client data, progress and attendance but cannot insert, update or delete the information.
* Caseworker can then sign out from this page

**An explanation of the mechanism you would use that enables inter-process communication in your application.**

**Jakob**

Inter-process-communication is a way for switching information amongst application process threads across many applications on different computers across a network on a single computer and the applications which take advantage of IPCs are referred to as a client server application.

**Alex**

Inter-process communication can allow a programmer to coordinate activities among different program process that can run very concurrently in an operating system. For this project we can use many different mechanisms to enable inter-process communication within our application but we decided on using .NET Remoting. Using .NET removing within our application allows us to build it to be widely distributed easily whether the application components are all on the one computer or are spread out across other computers possibly across the world.

**An explanation of what is an Interface in object-oriented programming and how would you use it to apply multiple inheritances.**

**Brayden**

An interface in object oriented programming is nothing like a user interface but instead is something that is very similar to a class but has no actual functionality and no actual code. The benefit to using an interface is that you can have different classes choose to implement the same interface while letting other parts of the app use objects with interfaces. Also using an interface will improve the efficiency of your code throughout loose coupling and abstraction.

**Jakob**

Multiple inheritance is when a class or object inherits features and properties from more than one parent class or object. Inheritances can and are implemented in a variety of ways depending on what language you are coding with. For Java including an inheritance for an interface requires you to type extends and then the name of the class you want to inherit from.

**Explain the concept of design patterns in the Java.**

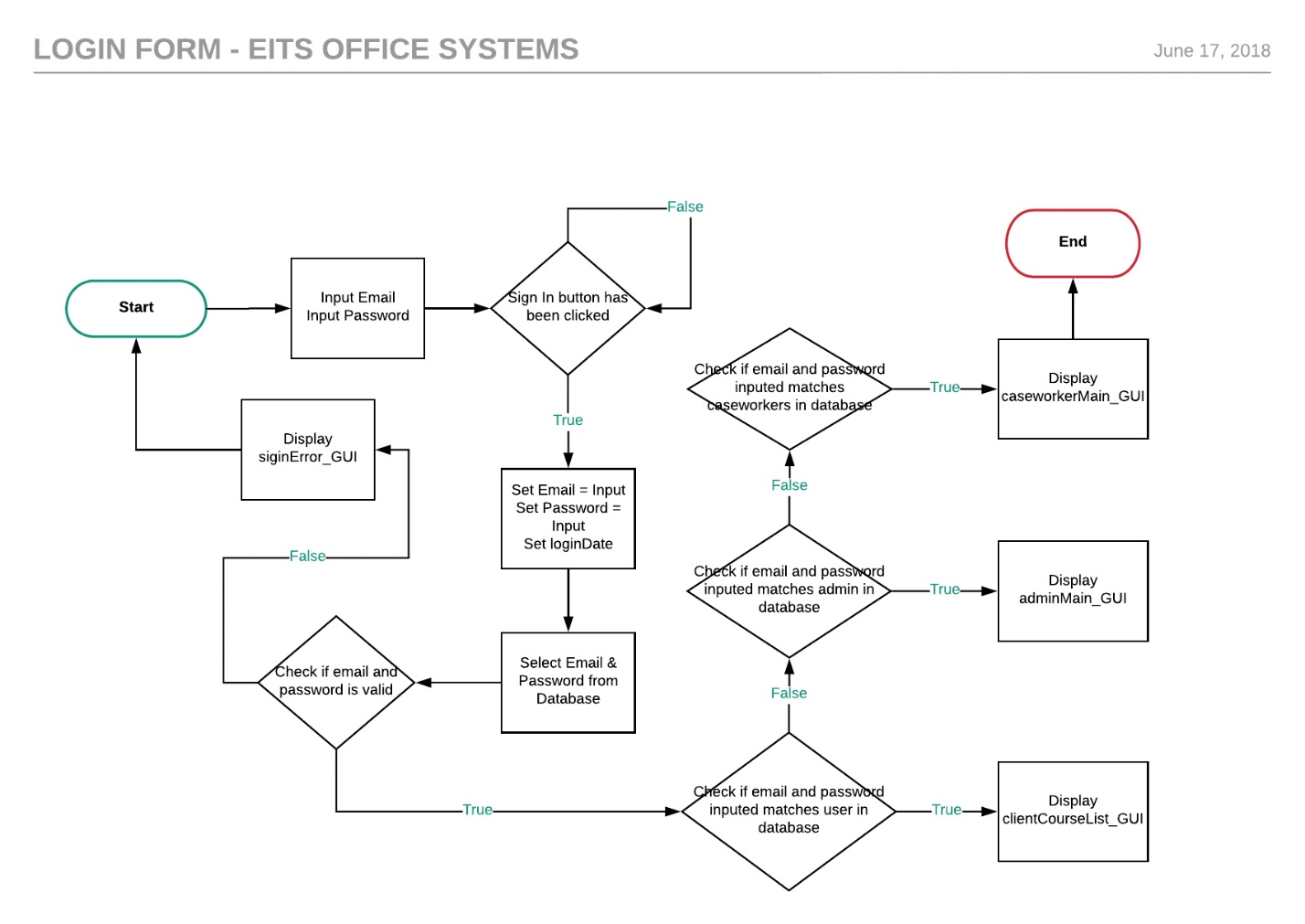
**Jakob**

Design patterns are practices that have been used to solve many design problems. Design patterns are used to help developers in producing higher quality software in a considerably faster amount of time. They are also not tied to any type of language or development platform.

**Alex**

Design patterns are fully reusable and lead to a more robust and much more maintainable code. It also makes for it to be easier to debug and for team members to understand it easily seeing as it usually similar to other things that are dealt with.

**Diagrams**

****

**Jakob**

Figure 26: Jakob - Completed Login Form Flowchart

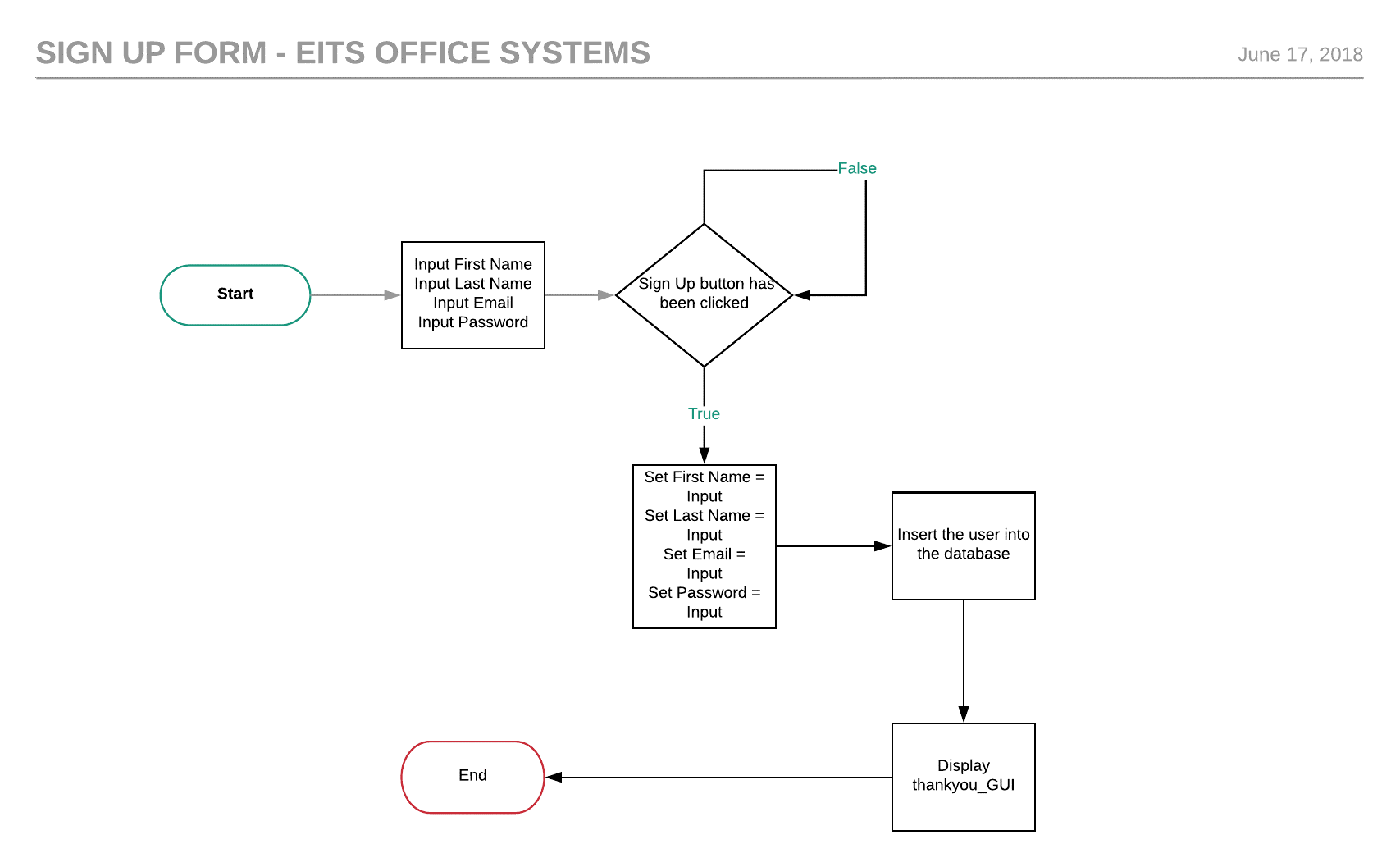
**Alex**

Figure 27: Alex - Completed Sign up form flowchart

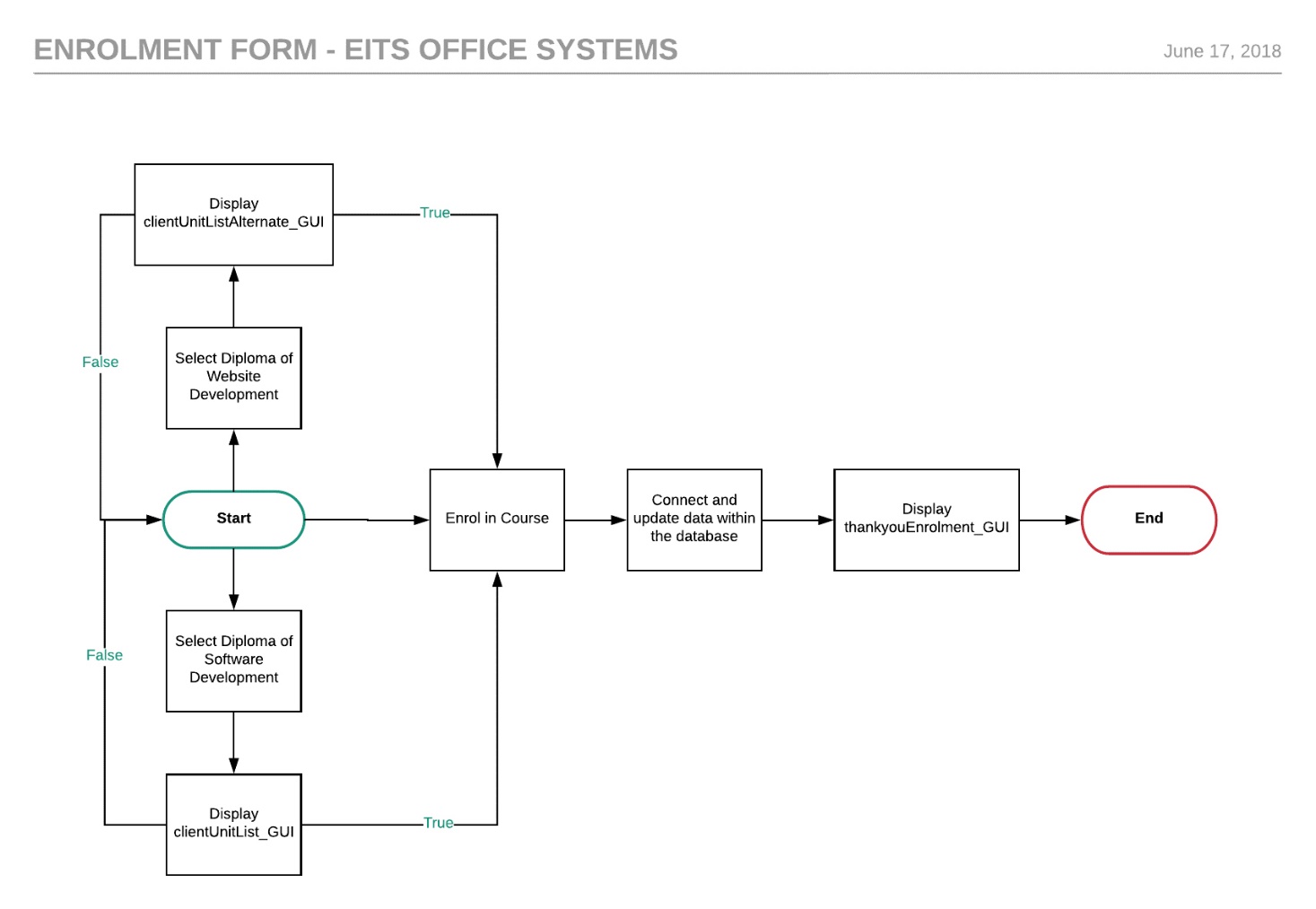
**Brayden**

Figure 28: Brayden - Completed Enrolment Form flowchart

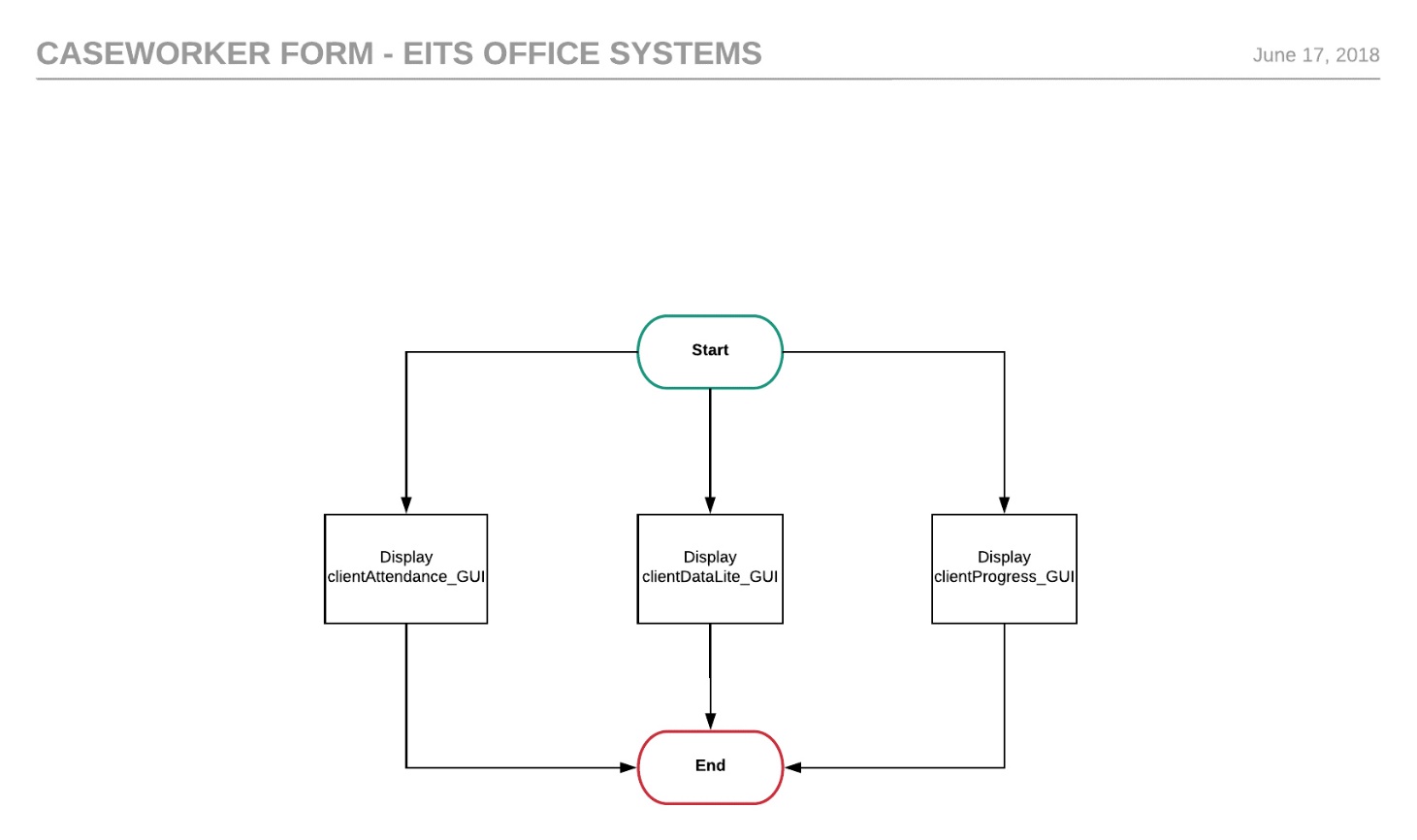
**Brayden**

Figure 29: Brayden - Completed Caseworker Form flowchart

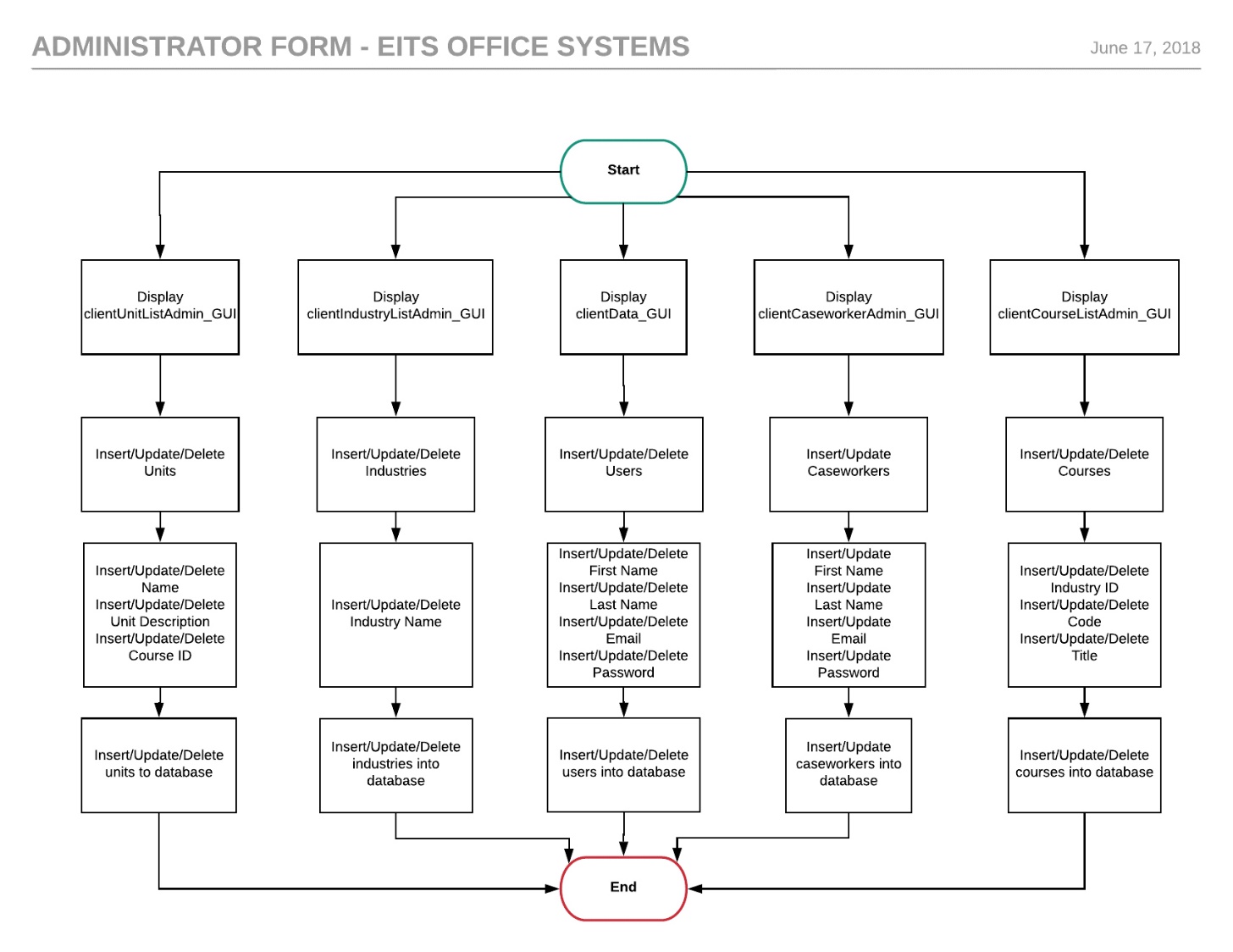
**Brayden**

Figure 30: Brayden - Completed Administrator Form flowchart

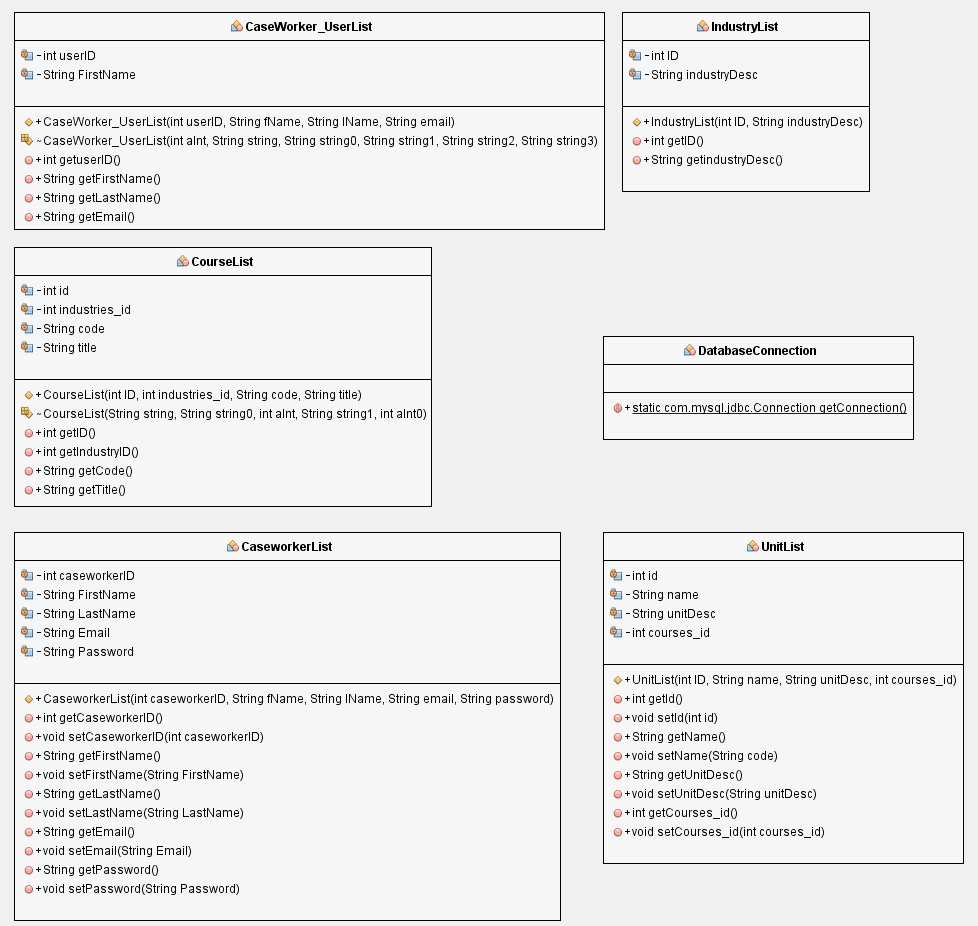
**UML Class Diagram:**

Figure 31: Finished UML Class Diagram 1



Figure 32: Finished UML Class Diagram 2

**TOE Charts:**

**Login – Jakob**

|  |  |  |
| --- | --- | --- |
| **Task** | **Object** | **Event** |
| Sign In to application | SignInButton | signupButtonActionPerformed |
| Sign Up to application | SignUpButton | signupButtonActionPerformed |
| Reset application | resetButton | resetButtonActionPerformed |
| Exit application | exitButton | exitButtonActionPerformed |
| Email Label | emailLabel |  |
| Password Label | passwordLabel |  |
| Welcome Label | welcomeLabel |  |
| Email Text Field | txtEmail |  |
| Password Text Field | txtPassword |  |

Table 3: Jakob - Login TOE chart table

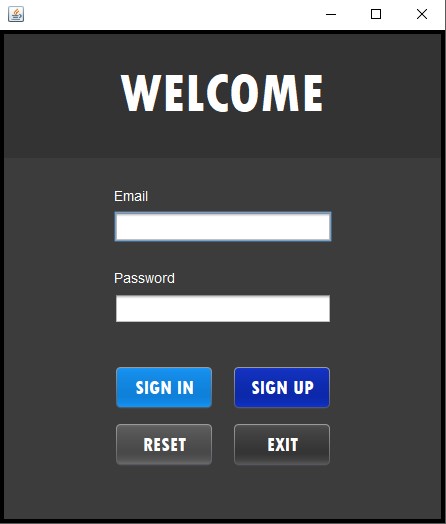
****

Figure 33: Jakob - Login Page TOE Chart

**Sign up – Jakob**

|  |  |  |
| --- | --- | --- |
| **Task** | **Object** | **Event** |
| Sign up to application | signupButton1 | signupButtonActionPerformed |
| Return to application | returnButton | returnButtonActionPerformed |
| Exit application | exitButton | exitButtonActionPerformed |
| First Name Label | firstnameLabel |  |
| Last Name Label | lastnameLabel |  |
| Email Label | emailLabel |  |
| Password Label | passwordLabel |  |
| Sign Up Label | signupLabel |  |
| First name Text field | txtFirstName |  |
| Last Name Text field | txtLastName |  |
| Email Text Field | TxtEmail |  |
| Password Text Field | txtPassword |  |

Table 4: Jakob - Sign up TOE chart table

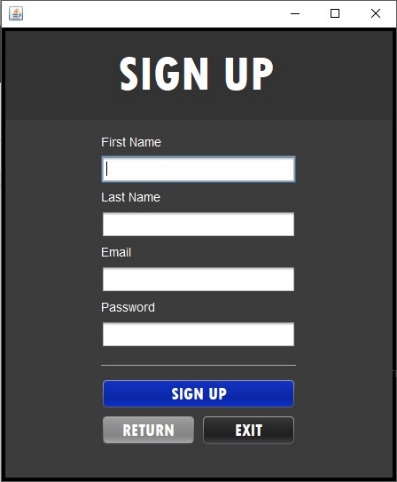
****

Figure 34: Jakob - Sign up page TOE chart

**Sign in error - Jakob**

|  |  |  |
| --- | --- | --- |
| **Task** | **Object** | **Event** |
| Back to application | backButton | backButtonMouseClicked |
| Exit application | exitButton | exitButtonMouseClicked |
| Error Label | errorLabel |  |
| Invalid email or password | invalidLabel |  |

Table 5: Jakob - Sign in error TOE chart table

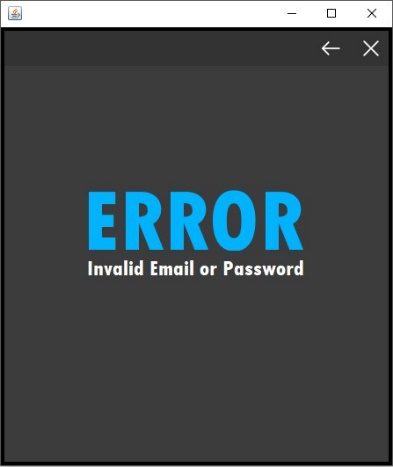
****

Figure 35: Jakob - Sign in error TOE chart

**Client Course List – Brayden:**

|  |  |  |
| --- | --- | --- |
| **Task** | **Object** | **Event** |
| Sign out to application | signoutButton | signoutButtonMouseClicked |
| Exit application | ExitButton | exitButtonMouseClicked |
| Diploma of software development to application | diplomaSoftwareDevButton | diplomaWebsiteDevButtonActionPerformed |
| Diploma of website development | diplomaWebsiteDevButton | diplomaWebsiteDevButtonActionPerformed |
| Information communication Label | InformationCommunicationLabel |  |
| Technology Label | technologyLabel |  |
| Course List | courseListLabel |  |
| Logged in as: | loggedInAsLabel |  |
| Bob | nameLabel |  |

****

Figure 36: Brayden - Client course list TOE chart

Table 6: Brayden - Client course list TOE chart table

**Client Unit List – Brayden:**

|  |  |  |
| --- | --- | --- |
| **Task** | **Object** | **Event** |
| Back to application | backButton | backButtonMouseClicked |
| Exit application | exitButton | exitButtonMouseClicked |
| Enrol to application | enrolButton | enrolButtonActionPerformed |
| Unit list | jListUnits |  |
| Diploma of software development Label | diplomaSoftwareDevLabel |  |
| Unit List Label | unitListLabel |  |
| Logged in as: | loggedInAsLabel |  |
| Bob | nameLabel |  |

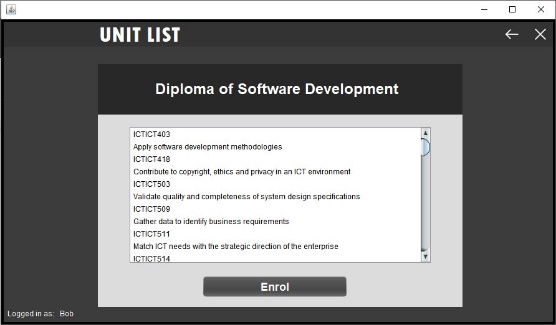
****

Figure 37: Brayden - Client unit list TOE chart

Table 7: Brayden - Client unit list TOE chart Table

**Client List Alternate – Jakob:**

|  |  |  |
| --- | --- | --- |
| **Task** | **Object** | **Event** |
| Back to application | backButton | backButtonMouseClicked |
| Exit application | exitButton | exitButtonMouseClicked |
| Enrol to application | enrolButton | enrolButtonActionPerformed |
| Unit list | jListUnits |  |
| Diploma of software development Label | diplomaWebsiteDevLabel |  |
| Unit List Label | unitlistLabel |  |
| Logged in as: | loggedInAsLabel |  |
| Bob | nameLabel |  |

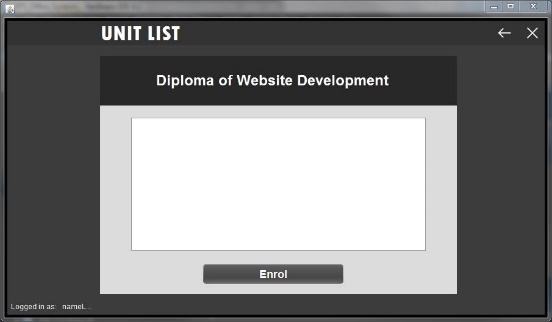
****

Figure 38: Jakob - Client list alternate TOE chart

Table 8: Jakob - Client list alternate TOE chart Table

**Admin Main – Brayden:**

|  |  |  |
| --- | --- | --- |
| **Task** | **Object** | **Event** |
| Client Data to application | ClientDataButton | ClientDataButtonActionPerformed |
| Caseworker Data to application | caseworkerDataButton | caseworkerDataButtonActionPerformed |
| Industry List to application | industryListButton | industryListButtonActionPerformed |
| Course List to application | courseListButton | courseListButtonActionPerformed |
| Unit List to application | unitListButton | unitListButtonActionPerformed |
| Sign Out to application | SignoutButton | SignoutButtonActionPerformed |
| Admin Panel Label | adminPanelLabel |  |

Table 9: Brayden - Admin main TOE chart Table

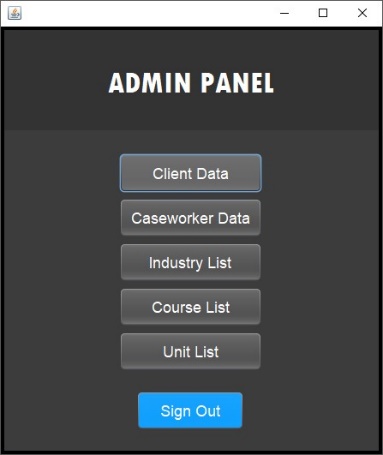
****

Figure 39: Brayden - Admin main TOE chart

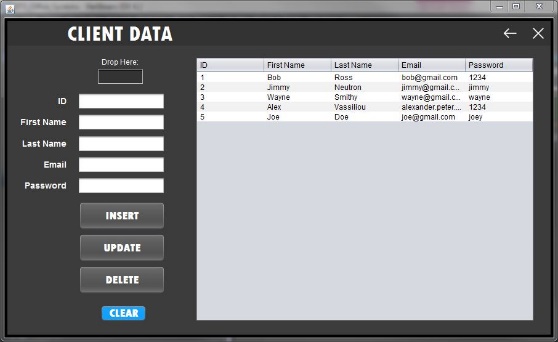
**Admin Client Data – Jakob:**

Figure 40: Jakob - Admin client data TOE chart

|  |  |  |
| --- | --- | --- |
| **Task** | **Object** | **Event** |
| Back to application | backButton2 | backButton2MouseClicked |
| Exit application | ExitButton | exitButtonMouseClicked |
| Insert to database | jInsertButton | jInsertButtonActionPerformed |
| Update to database | jUpdateButton | jUpdateButtonActionPerformed |
| Delete to database | jDeleteButton | jDeleteButtonActionPerformed |
| Clear to application | clearButton1 | clearButton1ActionPerformed |
| ID label | idLabel |  |
| First Name Label | firstnameLabel |  |
| Last Name label | lastnameLabel |  |
| Email Label | emailLabel |  |
| Password Label | passwordLabel |  |
| Drop Here Label | drophereLabel |  |
| Client Data Label | clientdataLabel |  |
| ID text field | jTextField\_ID |  |
| First name text field | jTextField\_FirstName |  |
| Last name text field | jTextField\_LastName |  |
| Email text field | jTextField\_Email |  |
| Password text field | jTextField\_Password |  |
| Drop here text field | jTextField\_drophere | jTextField\_drophereMouseEntered |
| Client Data Table | clientdataTable |  |

Table 10: Jakob - Admin client data TOE chart Table

**Admin Caseworker Data - Brayden:**

|  |  |  |
| --- | --- | --- |
| **Task** | **Object** | **Event** |
| Back to application | backButton2 | backButton2MouseClicked |
| Exit application | exitButton | exitButtonMouseClicked |
| Insert to database | jInsertButton | jInsertButtonActionPerformed |
| Update to database | jUpdateButton | jUpdateButtonActionPerformed |
| Clear to application | clearButton1 | clearButton1ActionPerformed |
| ID label | idLabel |  |
| First Name Label | firstnameLabel |  |
| Last Name label | lastnameLabel |  |
| Email Label | emailLabel |  |
| Password Label | passwordLabel |  |
| Caseworker Data Label |  |  |
| ID text field | jTextField\_ID |  |
| First name text field | jTextField\_FirstName |  |
| Last name text field | jTextField\_LastName |  |
| Email text field | jTextField\_Email |  |
| Password text field | jTextField\_Password |  |
| Caseworker data table | jDisplayCaseworkerTable |  |

Table 11: Brayden - Admin caseworker data TOE chart Table

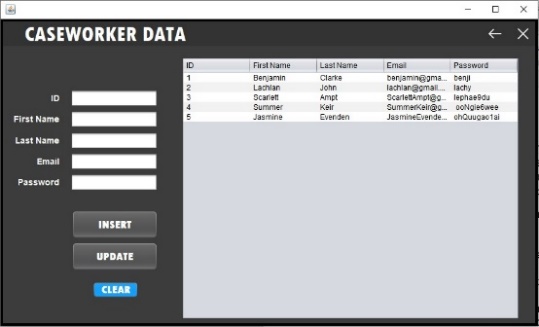
**  
Admin Industry List - Brayden:**

Figure 41: Brayden - Admin Caseworker Data TOE chart

|  |  |  |
| --- | --- | --- |
| **Task** | **Object** | **Event** |
| Back to application | backButton2 | backButton2MouseClicked |
| Exit application | ExitButton | exitButtonMouseClicked |
| Insert to database | jInsertButton | jInsertButtonActionPerformed |
| Update to database | jUpdateButton | jUpdateButtonActionPerformed |
| Delete to database | jDeleteButton | jDeleteButtonActionPerformed |
| Clear to application | clearButton1 | clearButton1ActionPerformed |
| ID label | idLabel |  |
| Industry Name Label | industryNameLabel |  |
| Industry List label | industryListLabel |  |
| ID text field | jTextField\_ID |  |
| Industry Name text field | jTextField\_IndustryName |  |
| Industry list table | jIndustryListTable |  |

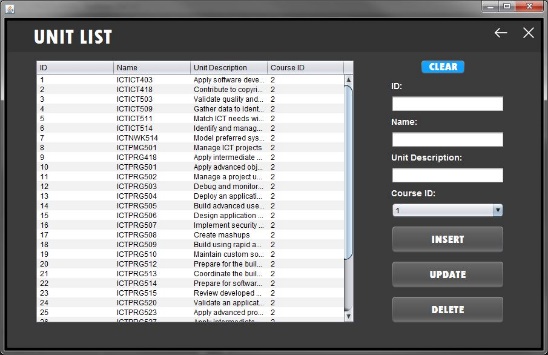
****

Figure 42: Brayden - Admin Industry list TOE chart

Table 12: Brayden - Admin industry list TOE chart Table

**Admin Course List - Brayden:**

|  |  |  |
| --- | --- | --- |
| **Task** | **Object** | **Event** |
| Back to application | backButton2 | backButton2MouseClicked |
| Exit application | ExitButton | exitButtonMouseClicked |
| Insert to database | jInsertButton | jInsertButtonActionPerformed |
| Update to database | jUpdateButton | jUpdateButtonActionPerformed |
| Delete to database | jDeleteButton | jDeleteButtonActionPerformed |
| Clear to application | clearButton1 | clearButton1ActionPerformed |
| Industry ID combo box | jSelectIndustryComboBox |  |
| Id label | idLabel |  |
| Industry ID label | industryLabel |  |
| Code label | codeLabel |  |
| Title Label | titleLabel |  |
| Course list label | CourselistAdmin |  |
| Id text field | jTextField\_ID |  |
| Code text field | jTextField\_Code |  |
| Title text field | jTextField\_Title |  |
| Course list table | jCourseListTable |  |

Table 13: Brayden - Admin course list TOE chart Table

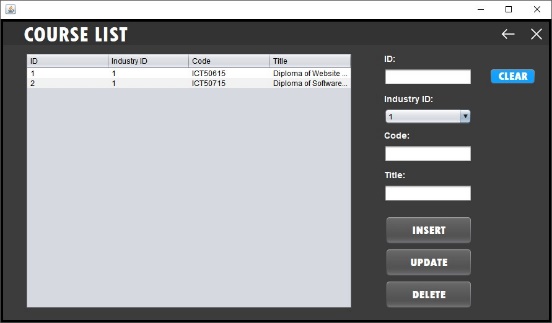
****

Figure 43: Brayden - Admin course list TOE chart

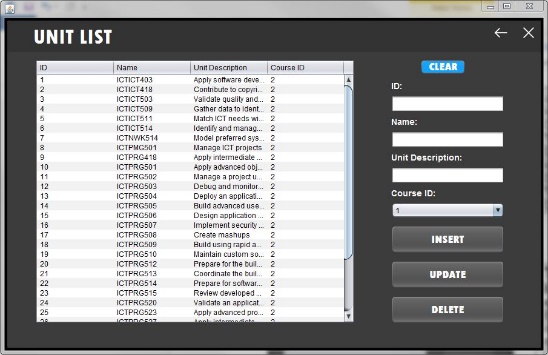
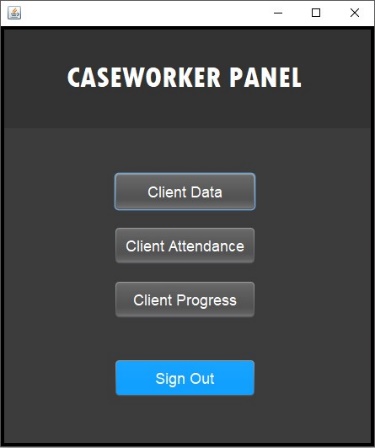
**Admin Unit List - Jakob:**

Figure 44: Jakob - Admin unit list TOE chart

|  |  |  |
| --- | --- | --- |
| **Task** | **Object** | **Event** |
| Back to application | backButton3 | backButton3MouseClicked |
| Exit application | ExitButton | exitButtonMouseClicked |
| Insert to database | jInsertButton | jInsertButtonActionPerformed |
| Update to database | jUpdateButton | jUpdateButtonActionPerformed |
| Delete to database | jDeleteButton | jDeleteButtonActionPerformed |
| Clear to application | clearButton1 | clearButton1ActionPerformed |
| Course ID combo box | jCourseComboBox |  |
| Id label | idLabel |  |
| Name label | nameLabel |  |
| Unit description label | unitDescriptionLabel |  |
| Course Id label | courseIDLabel |  |
| Unit list label | unitlistLabel |  |
| Id text field | jTextField\_ID |  |
| Name text field | jTextField\_Name |  |
| Unit description text field | jTextField\_UnitDesc |  |
| Unit list table | jUnitListTable |  |

Table 14: Jakob - Admin unit list TOE chart Table

**Caseworker Main - Alex:**

|  |  |  |
| --- | --- | --- |
| **Task** | **Object** | **Event** |
| Sign out to application | SignoutButton | SignoutButtonActionPerformed |
| Client Data to application | clientDataButton | clientDataButtonActionPerformed |
| Client attendance to application | clientAttendanceButton | clientAttendanceButtonActionPerformed |
| Client progress to application | clientProgressButton | clientProgressButtonActionPerformed |
| Caseworker panel label | caseworkerpanelLabel |  |

Table 15: Alex - Caseworker panel TOE chart Table

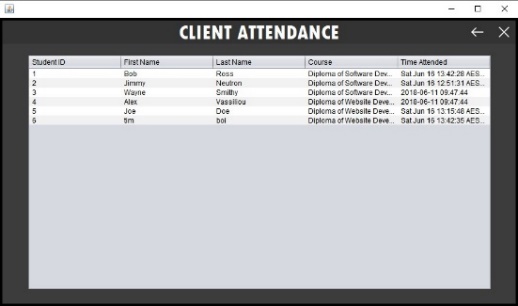
Figure 45: Alex - Caseworker main TOE chart

**Caseworker Client Data - Alex:**

|  |  |  |
| --- | --- | --- |
| **Task** | **Object** | **Event** |
| Back to application | backButton2 | backButton2MouseClicked |
| Exit application | ExitButton | exitButtonMouseClicked |
| Client Data Label | clientDataLabel |  |
| Search text field | jDisplayData | jDisplayDataKeyReleased |
| Client data table | jClientDataTable1 |  |

Table 16: Alex - Caseworker client data TOE chart Table

Figure 46: Alex - Caseworker client data TOE chart

**Caseworker Client Attendance - Alex:**

|  |  |  |
| --- | --- | --- |
| **Task** | **Object** | **Event** |
| Back to application | backButton2 | backButton2MouseClicked |
| Exit application | ExitButton | exitButtonMouseClicked |
| Client Attendance label | clientattendanceLabel |  |
| Client attendance table | jAttendanceTable |  |

Table 17: Alex - Caseworker client attendance TOE chart Table

Figure 47: Alex - Caseworker client attendance TOE chart

**Caseworker Client Progress - Alex:**

|  |  |  |
| --- | --- | --- |
| **Task** | **Object** | **Event** |
| Back to application | backButton2 | backButton2MouseClicked |
| Exit application | ExitButton | exitButtonMouseClicked |
| Client Progress Label | clientprogressLabel |  |
| Client progress table | jProgressTable |  |

Table 18: Alex - Caseworker client progress TOE chart Table

****

Figure 48: Alex - Caseworker client progress TOE chart

**Thank you Enrolment - Alex:**

|  |  |  |
| --- | --- | --- |
| **Task** | **Object** | **Event** |
| Back to application | backButton2 | backButton2MouseClicked |
| Exit application | ExitButton | exitButtonMouseClicked |
| Thank you label | thankyouLabel |  |
| For enrolling Label | forEnrollingLabel |  |
| You will hear back from us shortly label | hearbackshortlyLabel |  |

Table 19: Alex - Thank you enrolment TOE chart Table

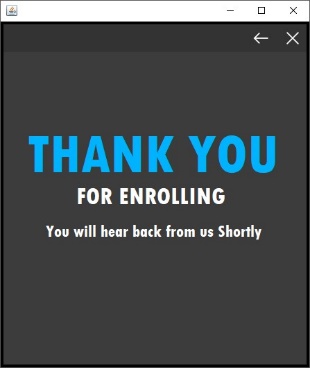
****

Figure 49: Alex - Thank you enrolment TOE chart

**Thank you for signing up – Alex:**

|  |  |  |
| --- | --- | --- |
| **Task** | **Object** | **Event** |
| Back to application | backButton2 | backButton2MouseClicked |
| Exit application | ExitButton | exitButtonMouseClicked |
| Thank you label | thankyouLabel |  |
| For signing up label | forSigningupLabel |  |
| You can now proceed to the sign in page label | proceedtosigninPageLabel |  |

Table 20: Alex - Thank you for signing up TOE chart Table

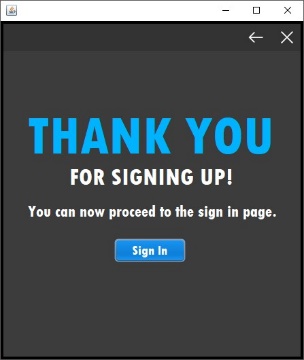
****

Figure 50: Thank you for signing up TOE chart

**Testing Performed:**

**Sign Up Interface: - Brayden**

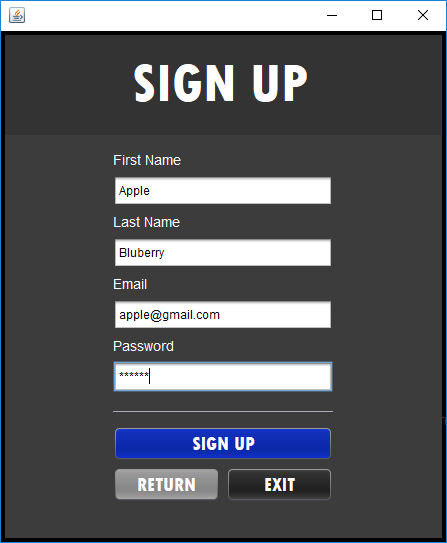
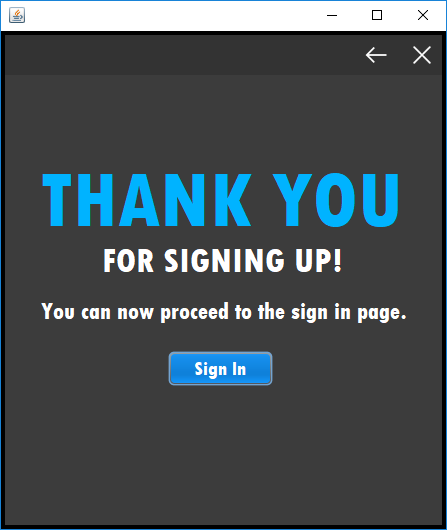


Figure 51: Brayden - Thank you for signing up GUI

Figure 52: Brayden - Sign up GUI

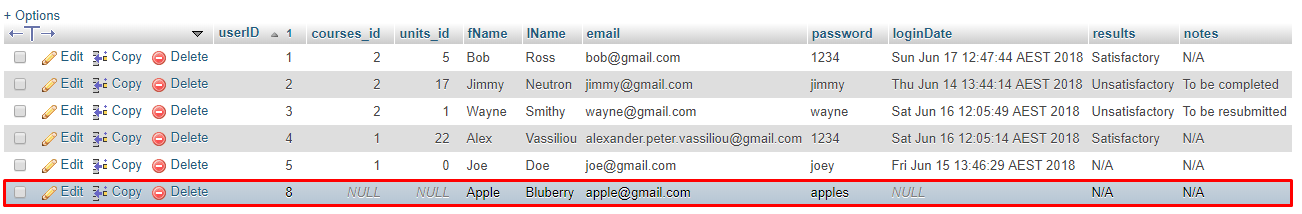


Figure 53: Brayden – Showing client added in Client Table within the Database

**Sign In Interface: Jakob**

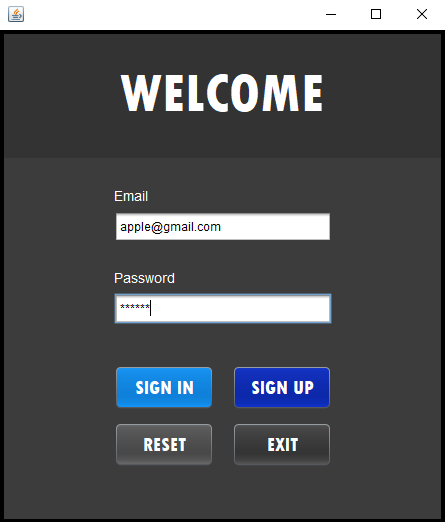


Figure 54: Jakob - Login page GUI

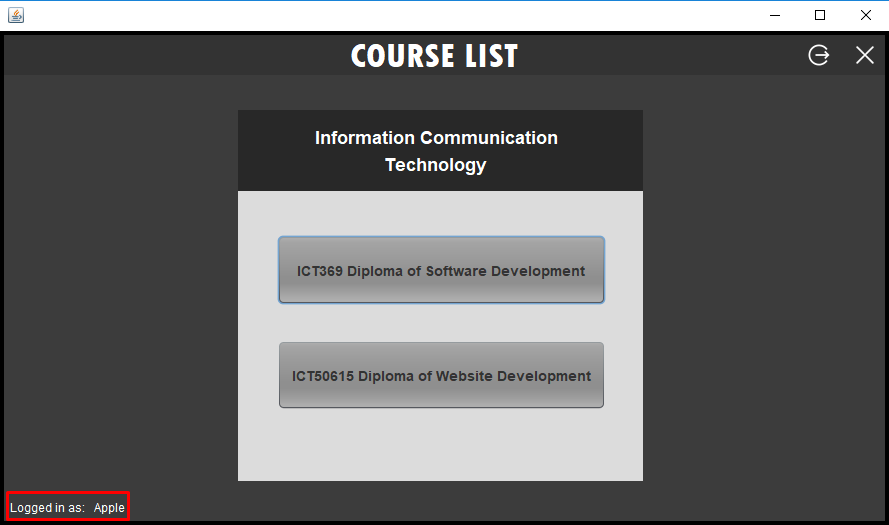


Figure 55: Jakob - Client Course List GUI

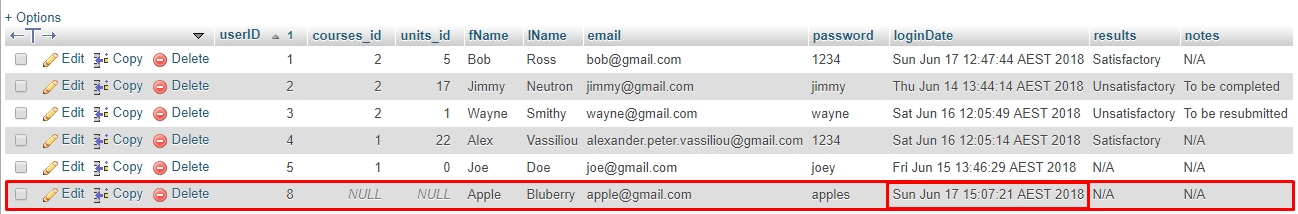


Figure 56: Jakob – Showing loginDate modified in Client Table within the Database

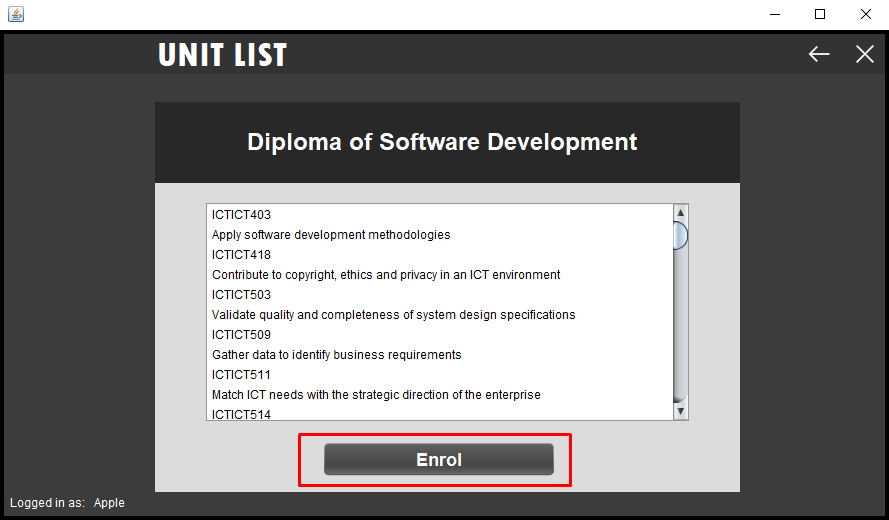
**Enrolment Interface: Alex**

Figure 57: Alex - Client unit List GUI

Figure 58: Alex - Client course list GUI

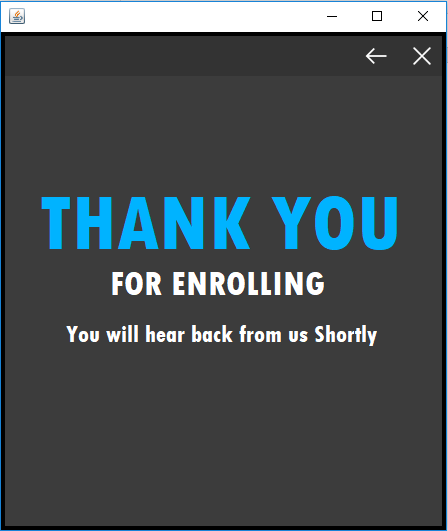


Figure 59: Alex - Thank you for enrolling GUI

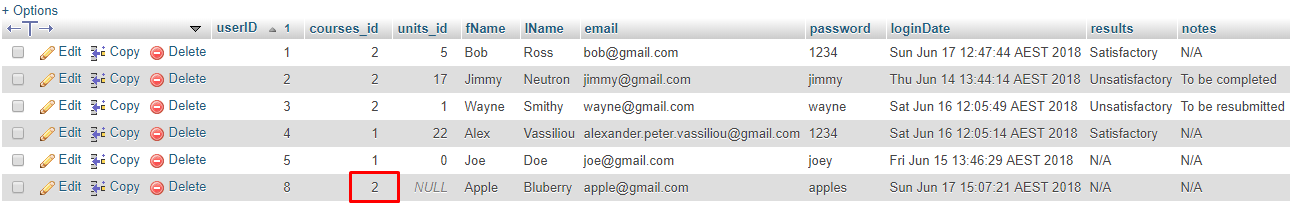


Figure 60: Alex - Showing Courses\_ID modified in Client Table within the Database

**Caseworker Interface: Brayden**

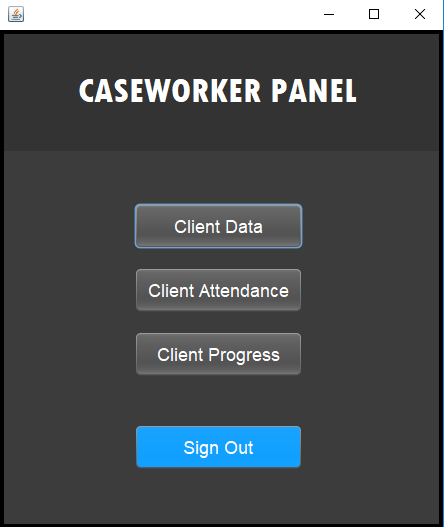
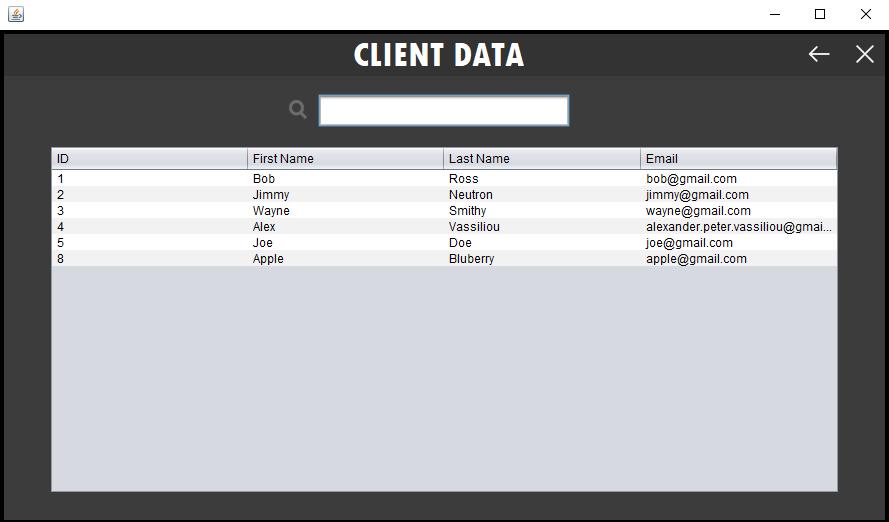
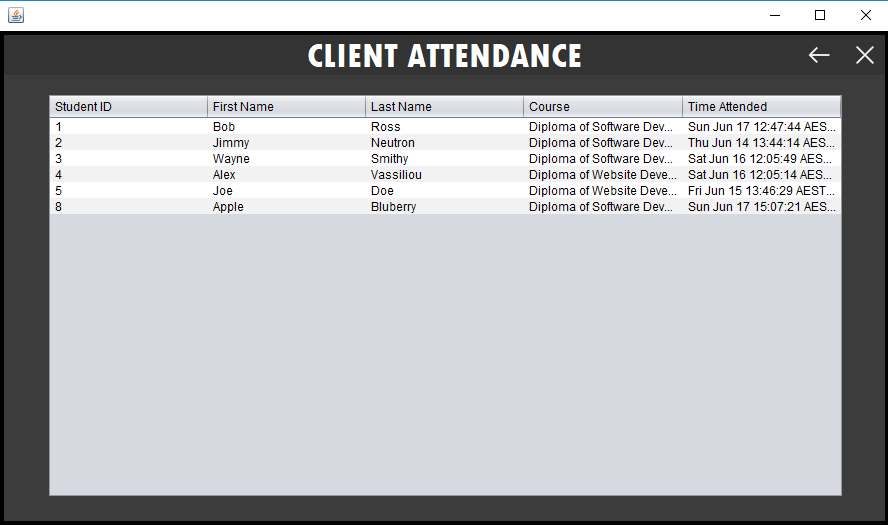


Figure 61: Brayden - Caseworker panel GUI

Figure 62: Brayden - Caseworker client data GUI

Figure 63: Brayden – Caseworker client Attendance GUI

Figure 64: Brayden - Caseworker client Progress GUI

**Admin Interface: - Alex**

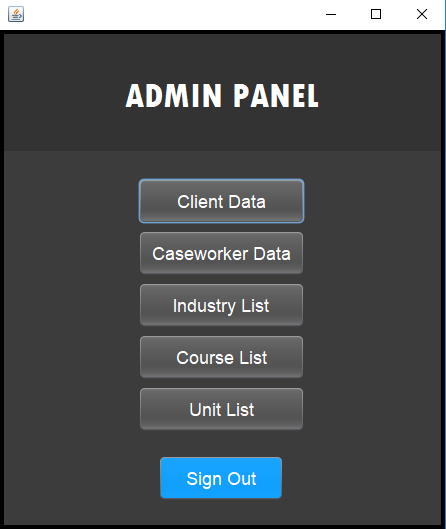


Figure 65: Alex - Admin panel GUI

**Client Data Insert/Update/Delete Test: - Brayden**

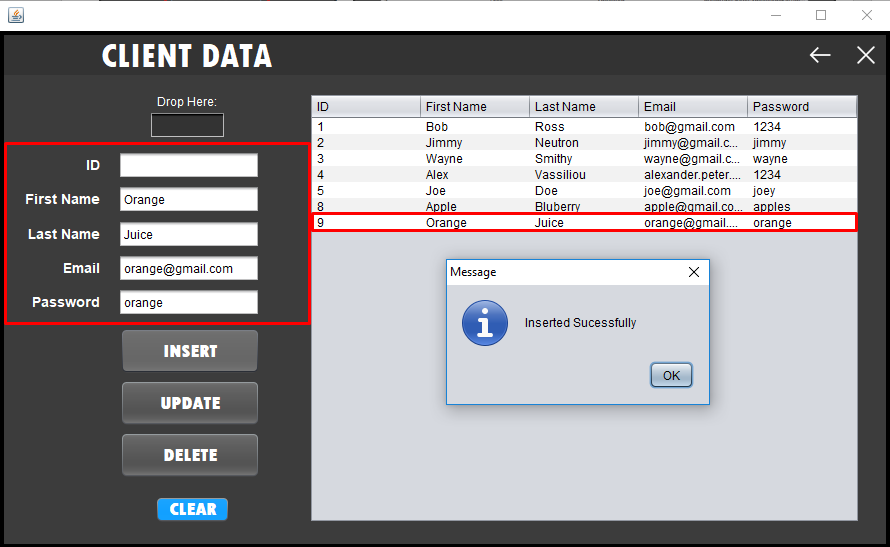
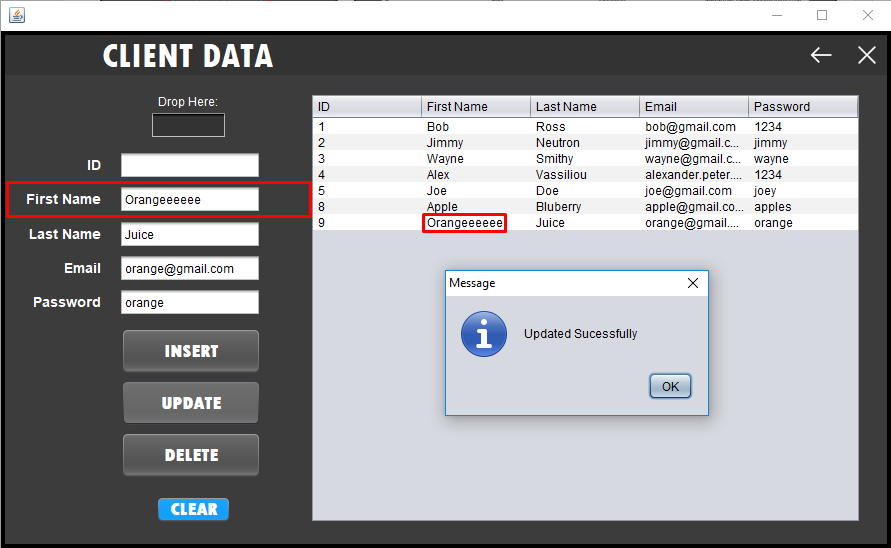


Figure 66: Alex - Demonstrating updating data in Admin client data GUI

Figure 67: Alex – Demonstrating inserting data in Admin client data GUI

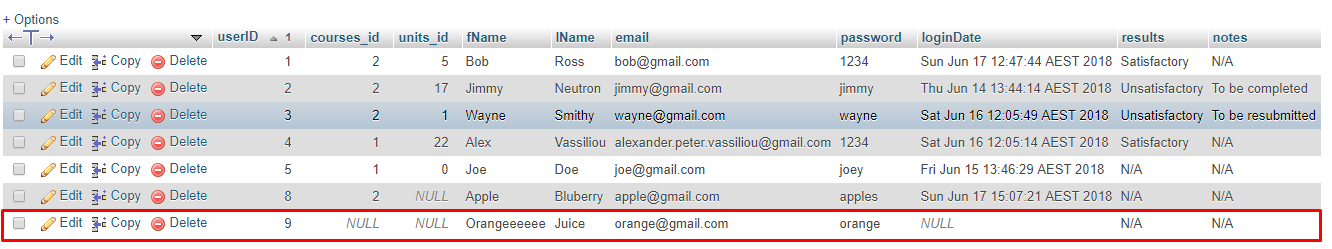


Figure 68: Alex - Showing Client added into Client Table within the Database

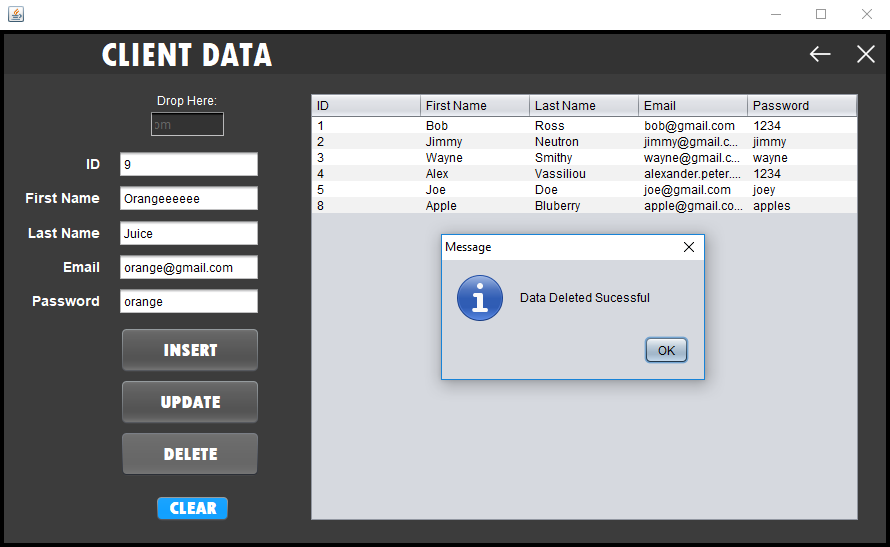


Figure 69: Brayden - Demonstrating Deleting Client in Admin client data GUI

**Caseworker Data Insert/Update Test: - Jakob**

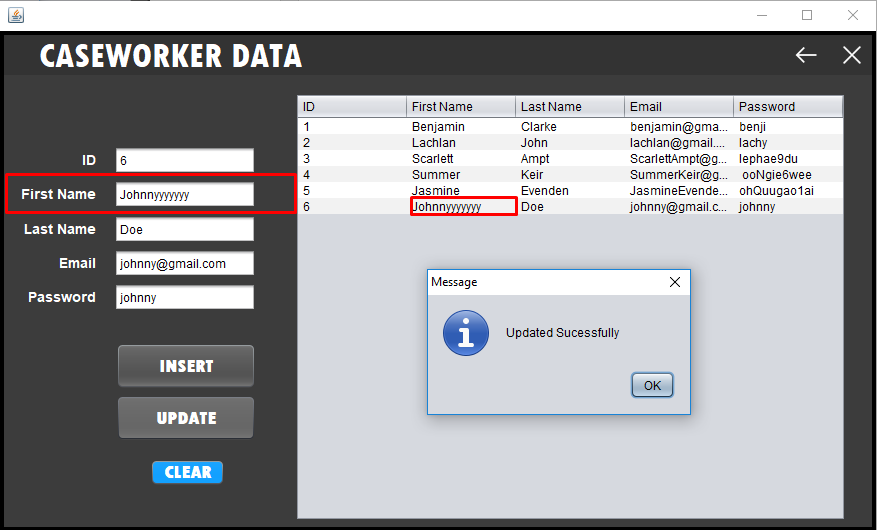
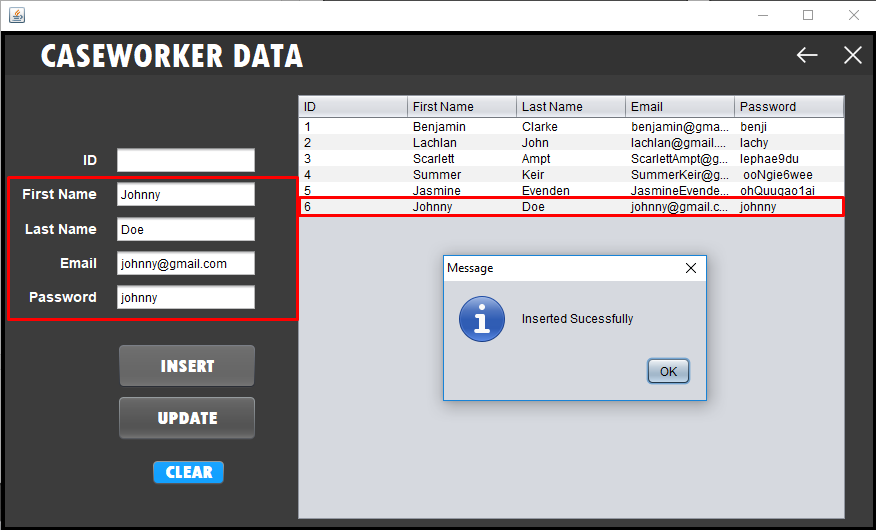


Figure 70: Jakob - Demonstrating inserting a caseworker in Admin caseworker data GUI

Figure 71: Jakob - Demonstrating updating a caseworkers name in Admin caseworker data GUI

**Industry List Insert/Update/Delete Test: - Alex**

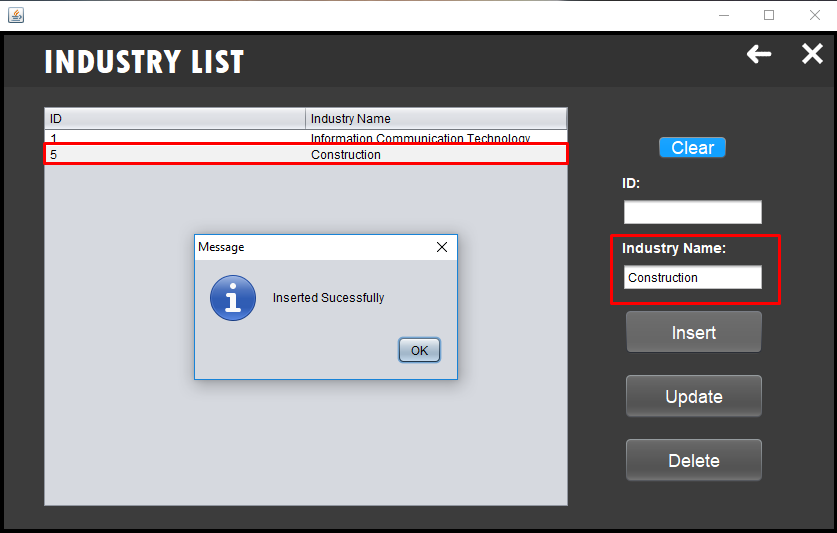
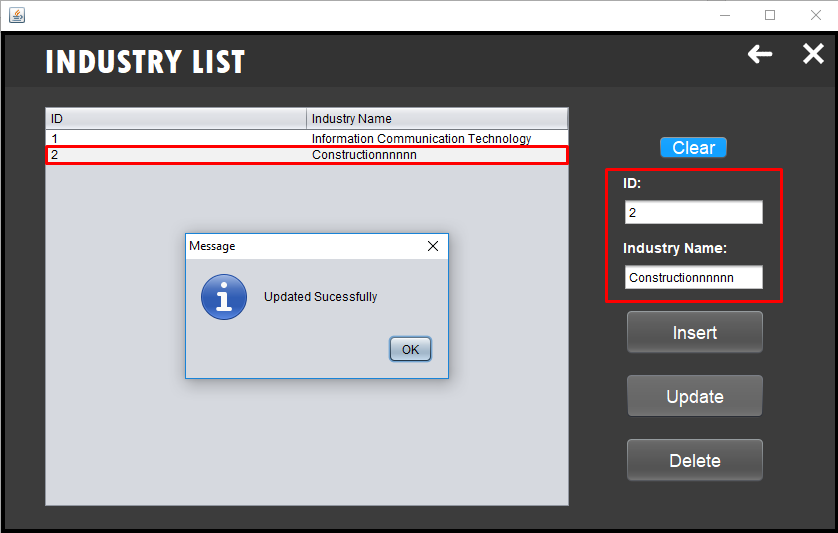


Figure 72: Alex - Demonstrating updating an industry in Admin industry list GUI

Figure 73: Alex - Demonstrating inserting an industry in Admin industry list GUI

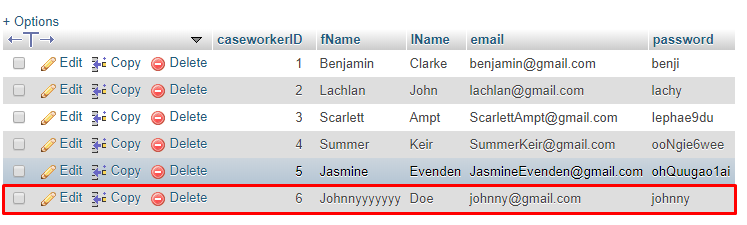


Figure 74: Alex - Showing client added into Caseworker table within the Database



Figure 75: Alex - Showing an industry added into Industry table within the Database

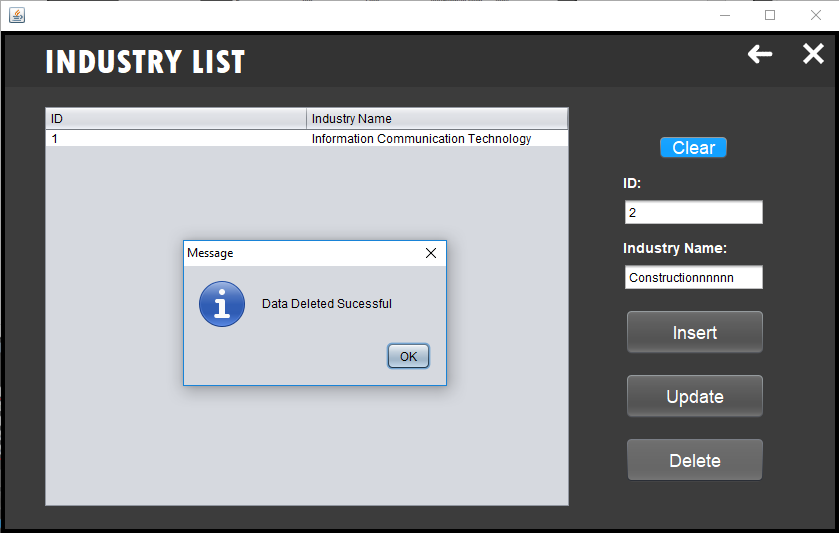


Figure 76: Alex - Demonstrating deleting an industry in Admin Industry list GUI

**Course List Insert/Update/Delete Test: - Brayden**

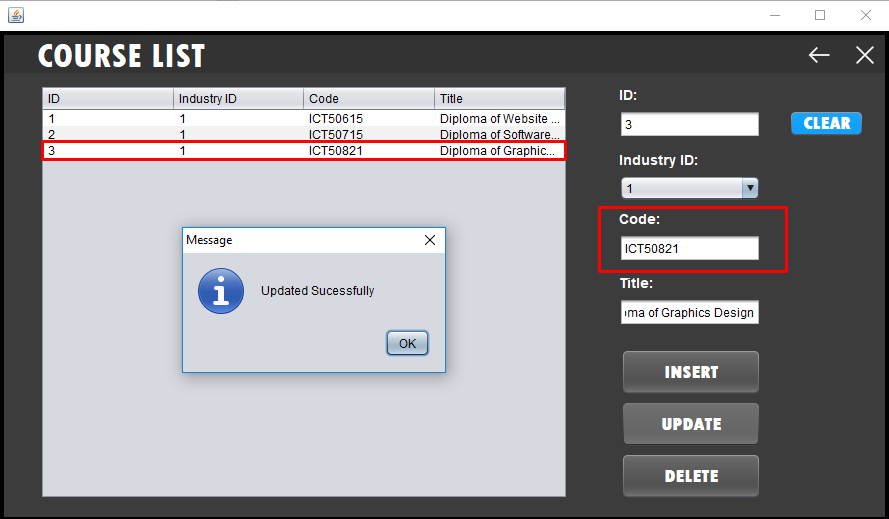
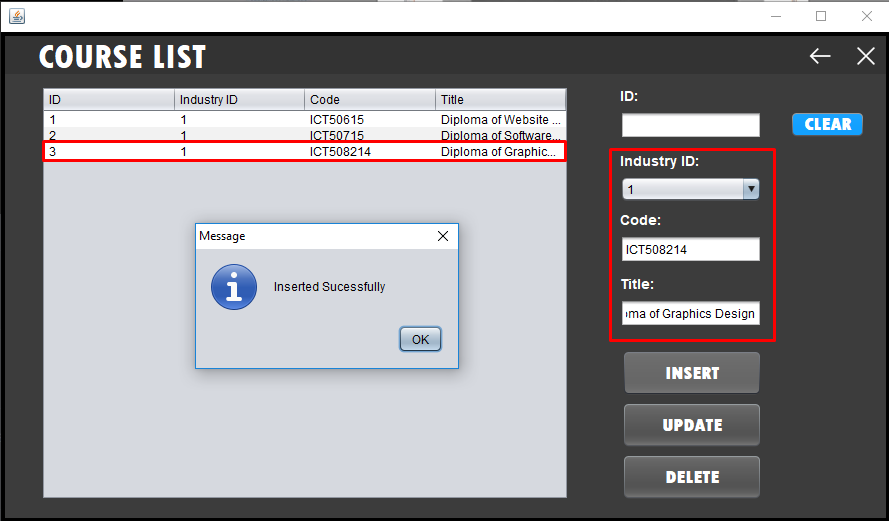


Figure 77: Brayden - Demonstrating inserting a course in Admin course list GUI

Figure 78: Brayden - Demonstrating updating a course in Admin course list GUI

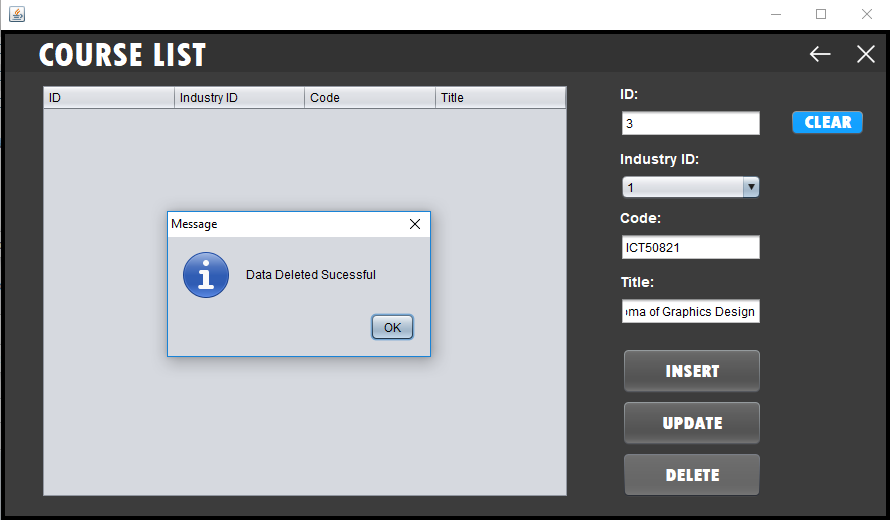


Figure 79: Brayden - Demonstrating deleting a course in Admin course list GUI

Figure 80: Brayden – Showing a Course added into Courses table within the Database

**Unit List Insert/Update/Delete Test: Brayden**

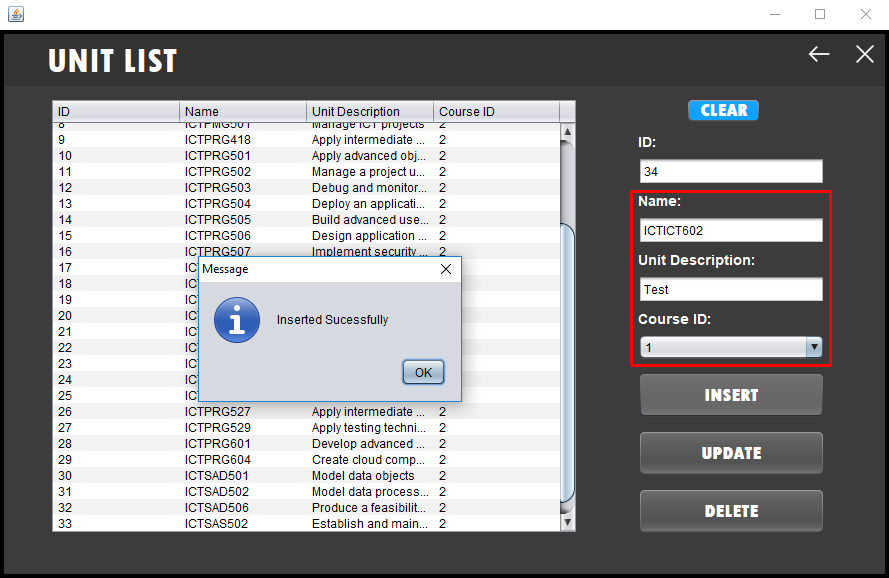
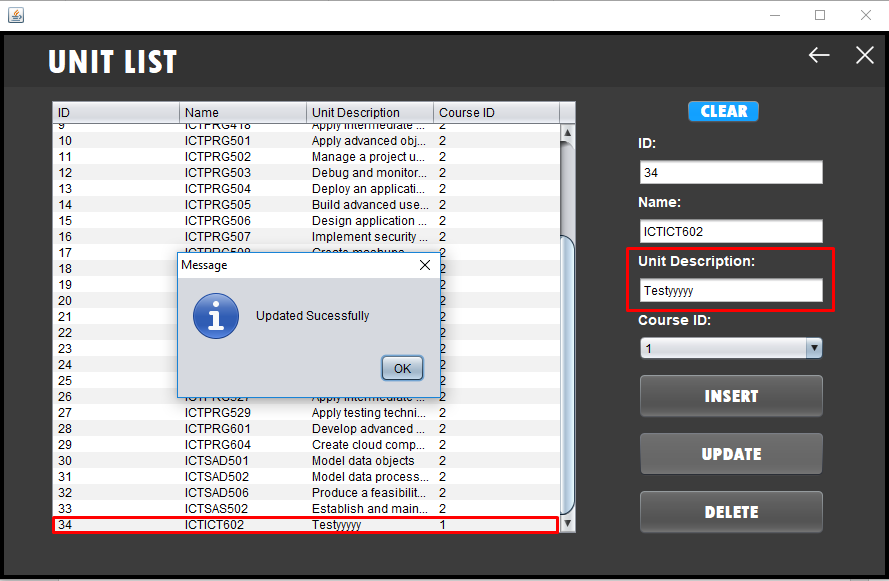
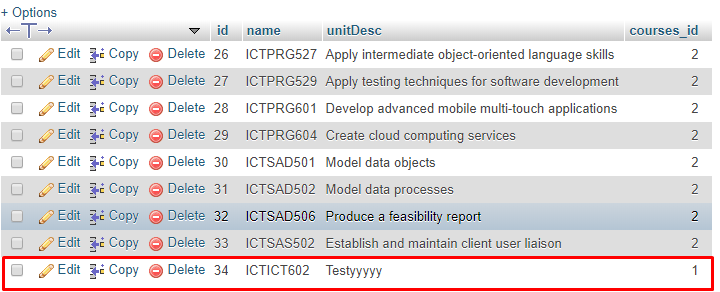


Figure 81: Brayden - Showing Unit added in Unit Table within the Database

Figure 82: Brayden - Demonstrating updating a unit in Admin unit list GUI

Figure 83: Brayden - Demonstrating Inserting a unit in Admin unit list GUI

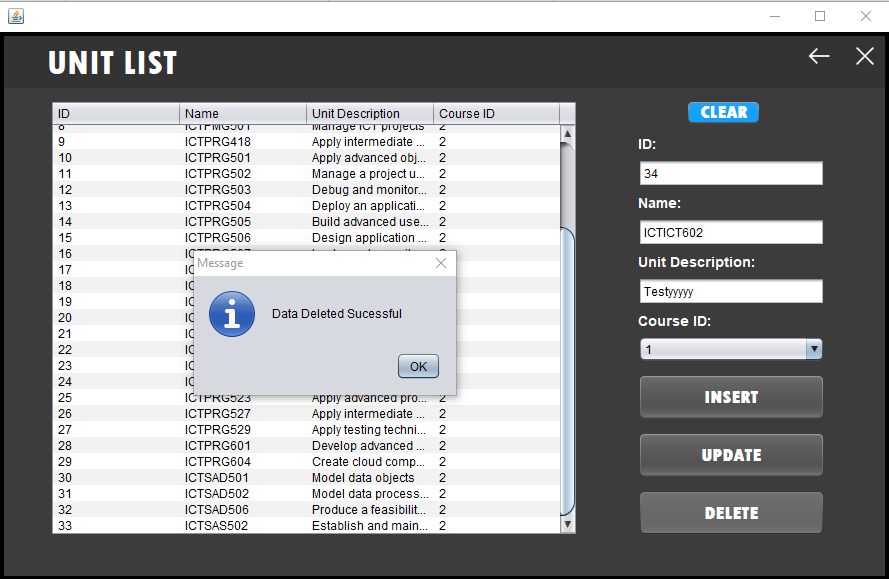


Figure 84: Demonstrating deleting a unit in Admin unit list GUI

**Overall Analysis of Test Results:**

**Brayden**

The overall analysis of the test results listed above performed very well during testing and comparing and meeting specific program specifications and gathered project requirements. It can be clearly displayed above that the functionality within the application during testing has been met well with the project requirements with only at least one bug occurring within an interface. The program specifications and gathered project requirements that have been met during the testing of each screen within the application include what the client ‘EITS’ has formally requested. Primarily `EITS` has requested a desktop application that allows clients/students to choose specific courses and track attendance and progress of these students.

Some examples of project requirements that have been met in relation to the test results listed above include the following:

* Welcoming clients to the office and tracking attendance – This was completed successfully during performing testing as it is clearly displayed above specifically for the interfaces of Sign in and Caseworkers. As a user logs in, their attendance is tracked throughout each form when beginning to select their specific course. This attendance is also shown towards caseworkers within the client attendance interface.

**Alex -**

* Administrators having the ability in inserting, updating, and deleting data across various tables within the database including client data, caseworker data, industries, courses and units. – This has been demonstrated above throughout the admin interface testing.
* Caseworkers obtaining the capability in displaying client data, client attendance and client progress. This functionality has been displayed above throughout the caseworker’s interface testing.

**Jakob**

In testing the course list for the administrators interface a minor bug occurred when deleting a specific course from the jTable and database. This bug consisted of deleting the specific course chosen although it allowed for the other courses included in the table to disappear. One way to resolve this issue is to refresh the specific interface which allows for the courses to appear again.

**Documentation Maintenance: - Jakob**

We maintained the program's documentation by implementing in author, purpose, version control, and date in comments at the top of each Form and Class. This will be used to help keep track of each form and class to easily allow people or yourself to come back to the code and understand the use and functions of the page.

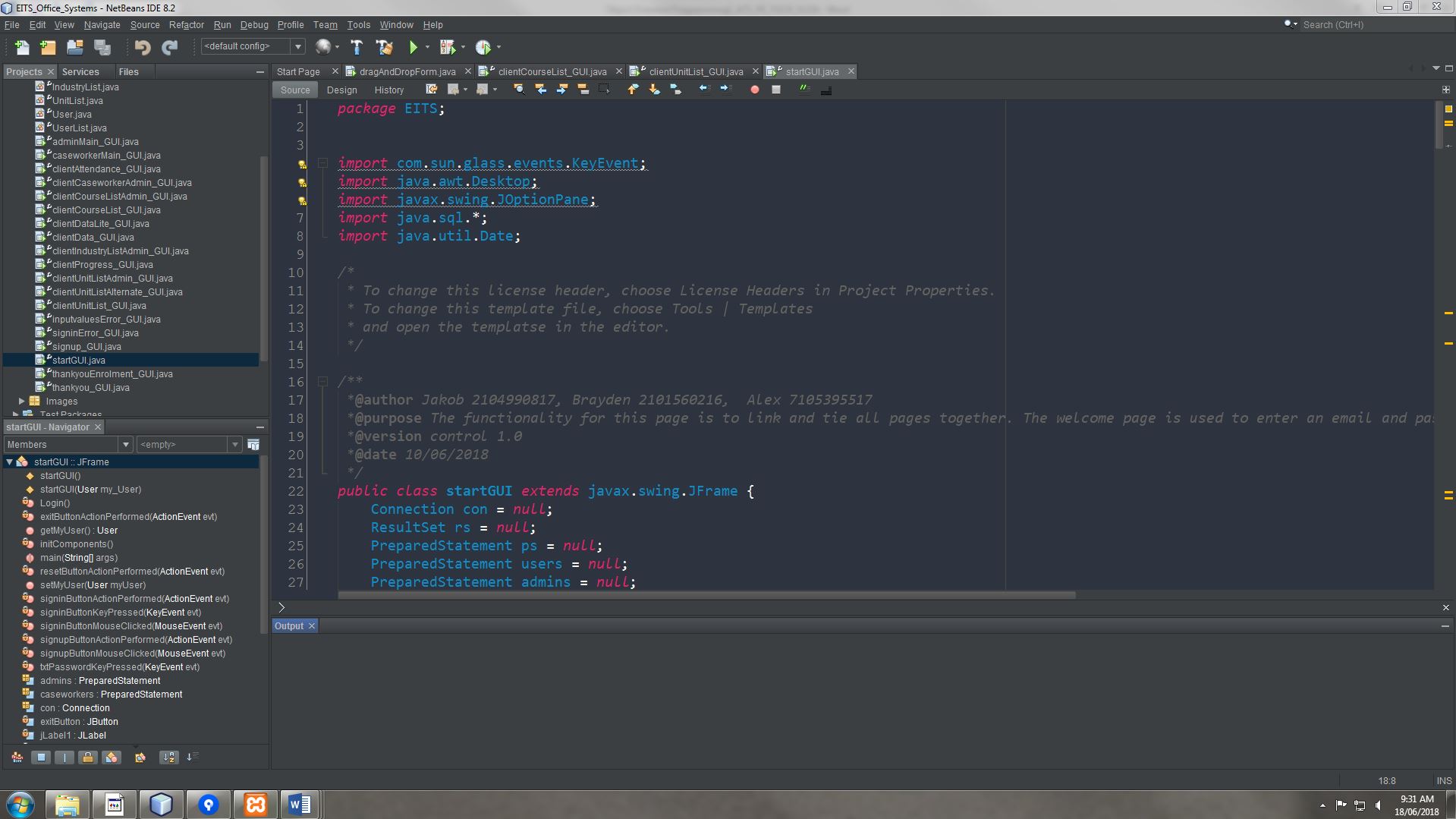
****

Figure 85: Jakob - Demonstrating use of programs documentation maintenance within startGUI

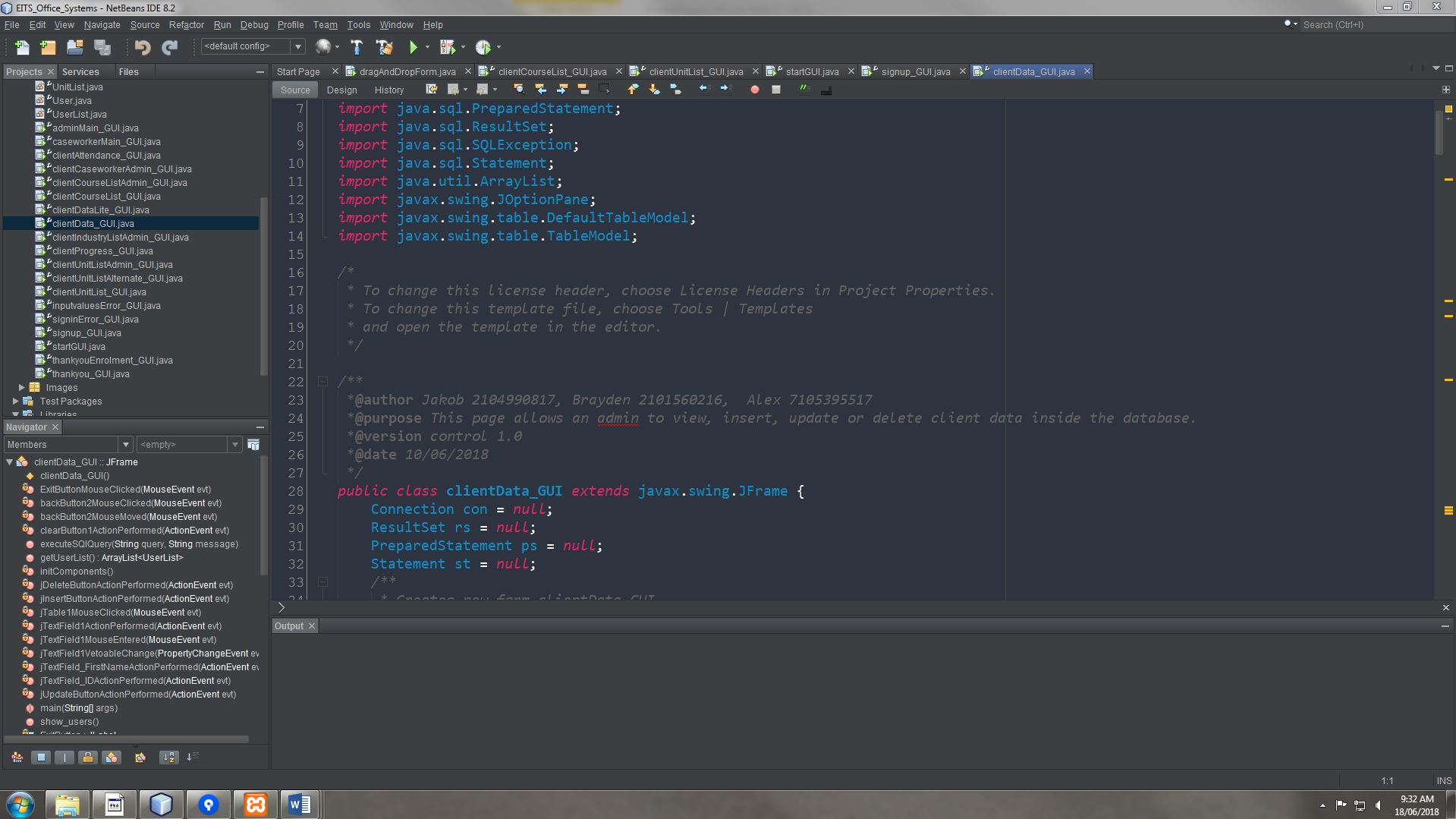
****

Figure 86: Jakob - Demonstrating use of programs documentation maintenance within clientData\_GUI

**Software Installation plan – Windows: Jakob:**

**Plan & Prepare**



You first want to plan and prepare to install a list of programs and to make sure they can operation with your operating system or hardware. It is recommended to make a backup of your system before the installation of the programs.

**Installation and System Requirements:**

* Windows 7, 8, or 10 machine 64 bit
* CPU with more than 2.5 GHz
* 4 GB of RAM
* 250+ GB of storage

**Programs and Files needed:**

* XAMPP Windows 7,8,10 64 bit version
* NetBeans Windows 7,8,10 64 bit version
* MySQL connector
* EITS-Office-Systems folder

**Programs recommended:**

* Sourcetree Windows 7,8,10 64 bit version

**Links:**

XAMPP: <https://www.apachefriends.org/download.html>

NetBeans: <https://netbeans.org/downloads/>

MySQL Connector: <https://dev.mysql.com/downloads/connector/j/5.1.html>

Sourcetree: <https://www.sourcetreeapp.com/>

**Install**

If your system passes the installation and system requirements you can begin installation of the programs needed and recommended to run the application.

Installation Guide: Locate the websites for each requested program and find the installation page. Locate the windows 7,8,10 64 bit version of the program and click download. Once your programs have been download begin installing the programs needed.

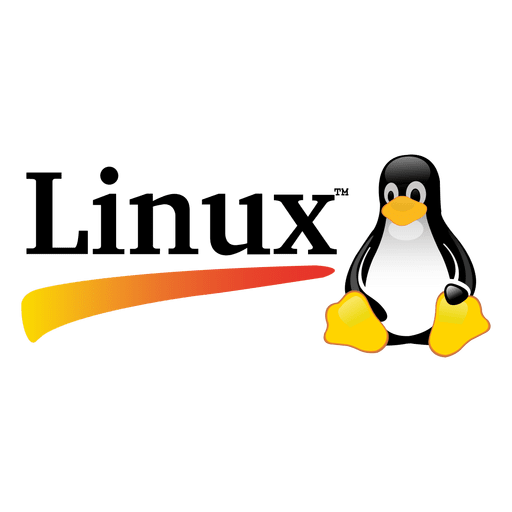
Go through the individual programs installations and once you have installed each program you can then move on to the next step.

**Configure**

1. Launch XAMPP and start MySQL and Apache
2. Click admin on MySQL within XAMPP and wait for the webpage to load
3. Now click New and Import which is located at the top
4. Import the eits\_office\_systems.sql to connect the application to the database
5. Launch NetBeans and open the EITS-Office-Systems folder
6. In NetBeans Library tab within projects add a JAR folder and open the MySQL-connector-java-bin.jar file
7. Now run the application within NetBeans

**Software Installation plan – Linux: Brayden:**

**Plan & Prepare**



Firstly, you need to plan and arrange to install a variety of programs and to make sure they can function appropriately with your operating system and hardware. It is recommended to make a backup of your system before the installation of the programs.

**Installation and System Requirements:**

* Any Linux machine 86/64 bit
* CPU with more than 2.5 GHz
* 4 GB of RAM
* 250+ GB of storage

**Programs and Files needed:**

* XAMPP for Linux 5.6.36, 7.0.30, 7.1.18 & 7.2. 64 bit version
* NetBeans Linux 86/64 bit version
* MySQL connector
* EITS-Office-Systems folder

**Programs recommended:**

* GitKraken Linux Ubuntu LTS 12.04 + Debian 8+ 86/64 bit version

**Links:**

XAMPP: <https://www.apachefriends.org/download.html>

NetBeans: <https://netbeans.org/downloads/>

MySQL Connector: <https://dev.mysql.com/downloads/connector/j/5.1.html>

GitKraken: <https://www.gitkraken.com/download>

**Install**

If your computer system obtains the installation and system requirements you can begin installation of the programs needed and recommended to run the application.

Installation Guide: Proceed to the websites for each program need and located the installation page. Find any of the Linux 86/64 bit versions of the program and click download. Once your programs have been downloaded begin installing the programs needed.

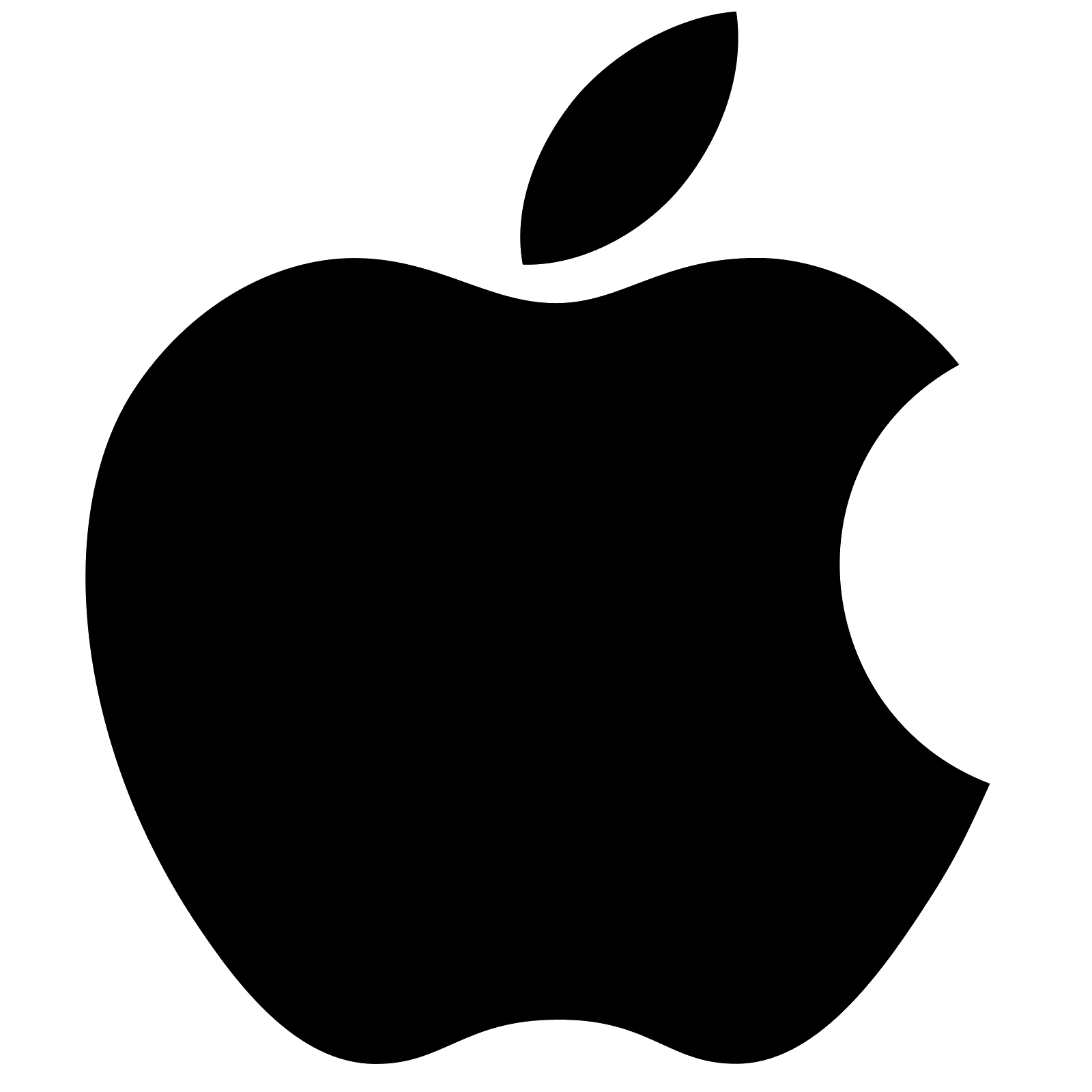
Go through the individual programs installations and once you have installed each program you can then proceed on to the next step.

**Configure**

1. Once the programs have been installed, launch XAMPP and start MySQL and Apache
2. Click admin on MySQL within XAMPP and wait for the webpage to load
3. Now click New and Import which is located at the top
4. Import the eits\_office\_systems.sql in order for the program to the connect to the database
5. Launch the program NetBeans and open the EITS-Office-Systems folder
6. In NetBeans Library tab within projects import a JAR folder and open the MySQL-connector-java-bin.jar file
7. Now run the application within NetBeans

**Software Installation plan – Linux: Alex:**

**Plan & Prepare**



You're first going to want to prepare a list of programs to install, then make sure that they can properly operate with your system and hardware. It's recommended to also make a backup before installing.

**Installation and System Requirements:**

* OS X 5.6.36, 7.0.30, 7.1.18, 7.2.6, 64 bit version
* CPU with more than 2.5Ghz
* 4 GB of RAM
* 250+ GB of storage

**Programs and Files needed:**

* XAMPP OS X 5.6.36, 7.0.30, 7.1.18, 7.2.6, 64 bit version XAMPP-VM & XAMPP-VM
* Netbeans OS X 5.6.36, 7.0.30, 7.1.18, 7.2.6, 64 bit version
* MySQL connector
* EITS-Office-Systems folder

**Programs recommended:**

* Sourcetree OS X 5.6.36, 7.0.30, 7.1.18, 7.2.6,64 bit version

**Links:**

XAMPP: <https://www.apachefriends.org/download.html>

NetBeans: <https://netbeans.org/downloads/>

MySQL Connector: <https://dev.mysql.com/downloads/connector/j/5.1.html>

Sourcetree: <https://www.sourcetreeapp.com/>

**Install**

If your system has the required hardware you can begin the installation on the programs.

To begin you will have to locate the version of OS X 5.6.36, 7.0.30, 7.1.18, 7.2.6,64 bit version any of them, and click the download button. Once your programs are finished download begin the installation of them.

Go through the program installations and once they are all installed continue to the next step.

**Configure**

1. Launch XAMPP through your finished download on mac, and start MySQL and Apache

2. Click admin on MySQL within XAMPP and wait for the webpage to load

3. Now click new and import which is located at the top

4. Import the Eits\_office\_systems\_sql to connect the application to the database

5. Launch netbeans and open the EITS-Office-Systems folder

6. In netbeans library tab within project add a JAR folder open the mysql connector java bin jar file

7. Now run application with netbeans

# **10.0 References**

Sourcetree - <https://en.wikipedia.org/wiki/Atlassian>

Sourcetree - <https://sagittarius.agency/blog/5-reasons-to-use-sourcetree-for-git>

Microsoft Project - <https://en.wikipedia.org/wiki/Microsoft_Project>

Microsoft Project - <https://www.mpug.com/education/what-is-microsoft-project/>

Jira - <https://en.wikipedia.org/wiki/Jira_(software)>

Jira - <https://www.atlassian.com/software/jira/agile>

GUI - <https://netbeans.org/kb/docs/java/gui-functionality.html>

GUI - <https://www.cse.wustl.edu~cytron/101Pages/swf12/HelpDocs/NetBeans/netbeanstutorial.htm>

GUI - <https://www.slideshare.net/ewinrosseneriaxl/netbeans-gui-tutorial>

Database - <https://netbeans.org/kb/docs/ide/java-db.html>

Database - <https://netbeans.org/kb/docs/ide/mysql.html>

Database - <https://www3.ntu.edu.sg/home/ehchua/programming/java/JDBC_Basic.html>

OTHERS -

<https://www.w3schools.com/>

<https://www.codecademy.com/learn/learn-java>

<https://netbeans.org/>