**Design of my App:**

The user interface for the app will be based on a whiteboard background to mimic using them in lessons. Text will be displayed in easy to read fonts and colours to make the app simple and pleasant to use. I will stick to only a couple of colours to keep the design consistent throughout which is one of the design principles which helps reduce user errors. Buttons will be used which will allow users to tap on the screen to use them and they will be sufficiently large enough so that users don’t accidently tap on the wrong part of the screen. I will use tables to display information such as leaderboard and group views which will allow the user to swipe up and down on them to go through the rows. I will not include back buttons as users will be able to use the android built in buttons as it will help keep the design uncluttered and simple.

The app will have a welcome screen which has two buttons-one for login and one for registration. Allowing users to create an account will make it easier to store progress such as high scores and edit questions that they created. The registration section will ask users to enter a username, password and an email address. The email address is required so that when a user forgets their login details they can recover them through an email. There will be checks to make sure that none of the fields are empty, then a connection will be made to the registration script which will store the details into a new row in the user table of the database. However if a username or email already exists then a message will be returned to the app which will be displayed to the user explaining that the username is in use or the email address already has a username attached.

For logging in a user can enter a username and password which will then be checked against the database user table to check if that user exists, they can also request an email to be sent to them to recover their details. I have chosen to send an email instead of displaying the details on screen because another user on a different device could find out someone else's email address and enter it which would then display that users details, whereas my solution will keep the details private as the user can only find them out by logging into their email server.

A home page will appear after a user successfully logs in and it will display the logged in user with their current highscore. Here the user can then choose one of three sections (create, play and leaderboard). The create and play sections are the main part of the app these are where the user can create quizzes and play them. I have added a leaderboard section as an extra feature to give some competition between users to give them another reason to use the app and motivate them more to revise using this app. The leaderboard will display the username and highscore of the top 50 players, logged in users will have their position highlighted if they are on the leaderboard to make it easier for them to see.

The create section will give the users the option to create a quiz or create a question or view current groups that they are in. I have split up the create a quiz and create a question to make it easier for groups/users to create new questions for existing quizzes by having less steps - as they won't have to go through the create quiz section each time. When creating a question users will be able to choose whether to create them on their own or as a group, one of the requirements of the project is for users to be able to create questions in a group as a collaborative effort. My solution will allow up to 3 other players (4 including question creator) to take part in a single question.

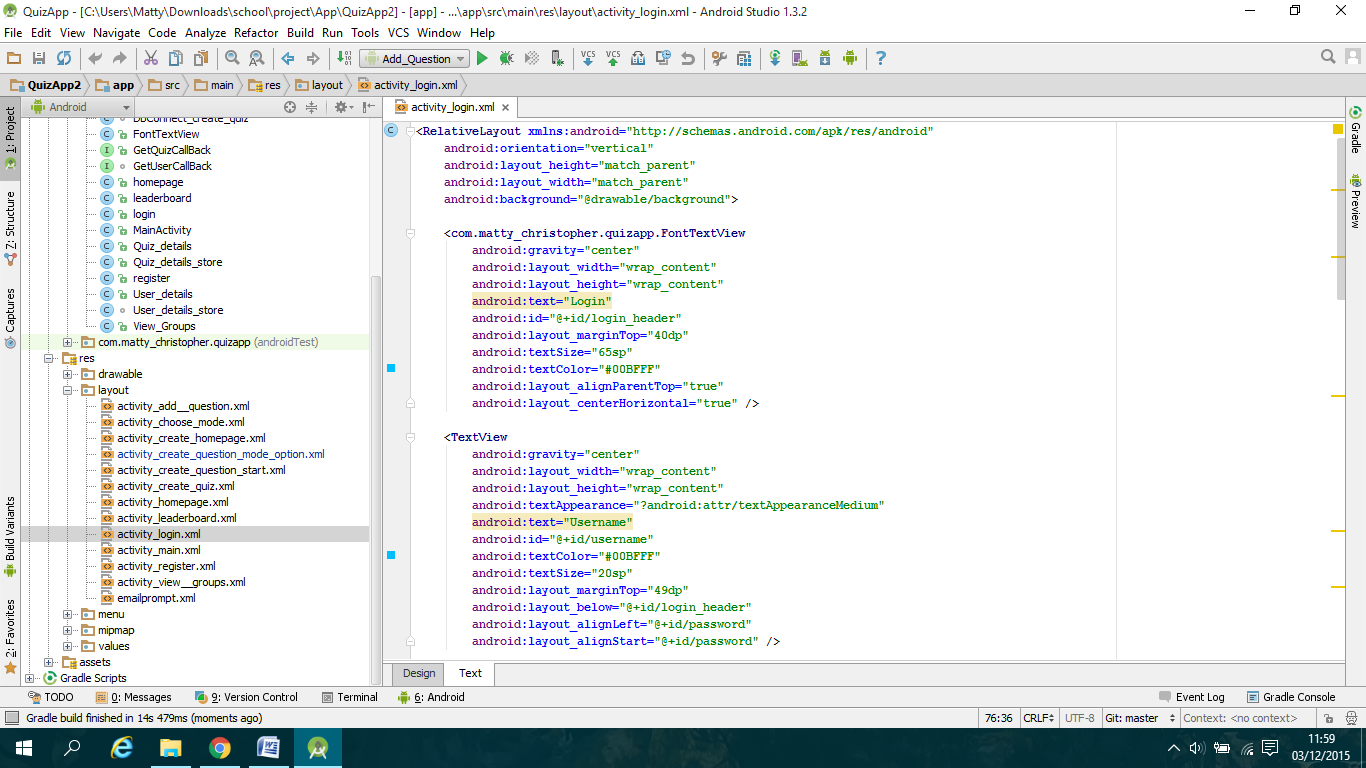
The question creator will be able to enter 3 other names they want to join to the group which will then be displayed in the view groups section. Users will be able to view the groups they are in then select one of the groups to edit the question where they will be able to edit the answer that they have been assigned to e.g. "false answer 1". Once all of a question are filled in then the creator will be able to send the question off as complete so it can be used for the play section.

An extra feature that I will include in the app is to allow users to play a highscore mode which chooses random questions from different topics and will keep going until user gets one wrong, this will help improve replay ability even after users don’t need to study anymore. Players will be able to rate questions after they have answered them to help ensure that better rated questions are more likely to appear, they will also be able to view stats for the questions which will display the percentage for each answer what users tended to select. This will help users see possible patterns for questions so that they won't make these mistakes in exams.

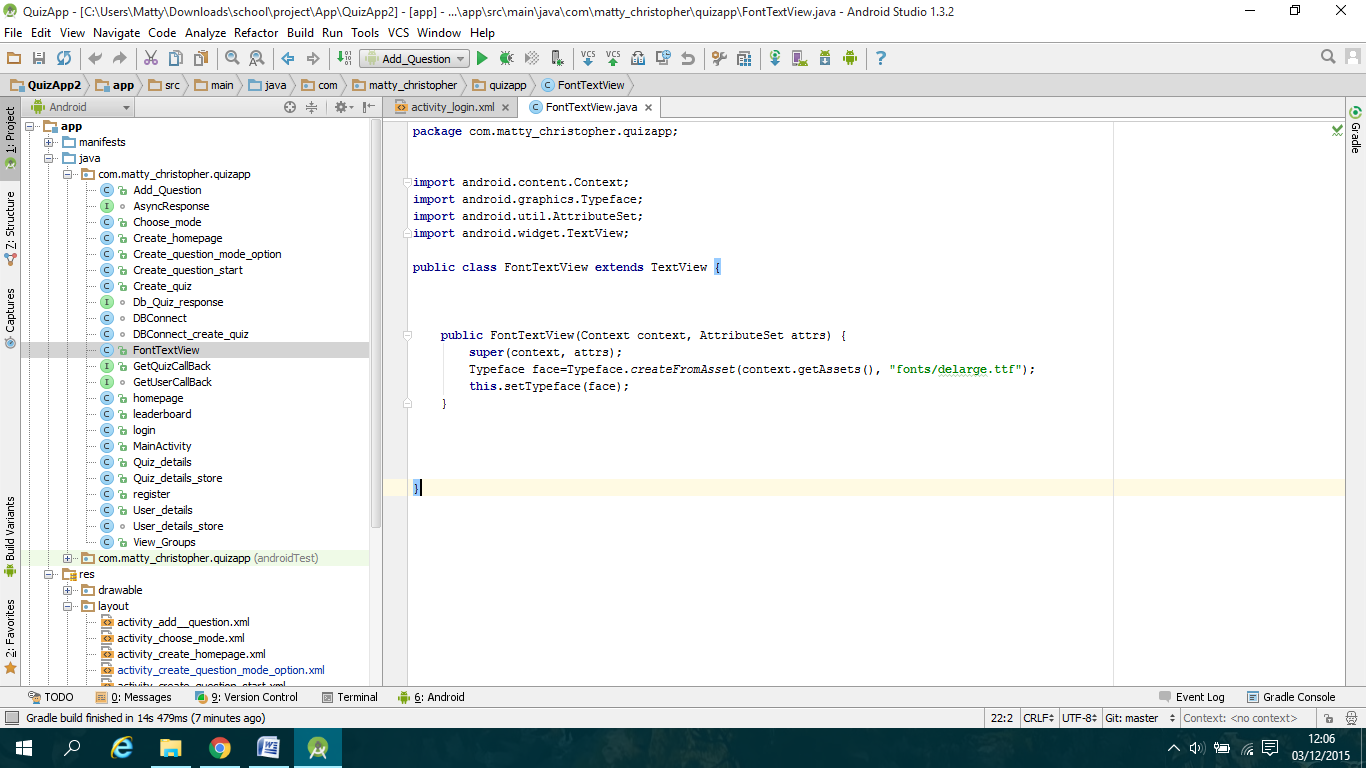
**Implementation:**

*Creating the user interfaces:*

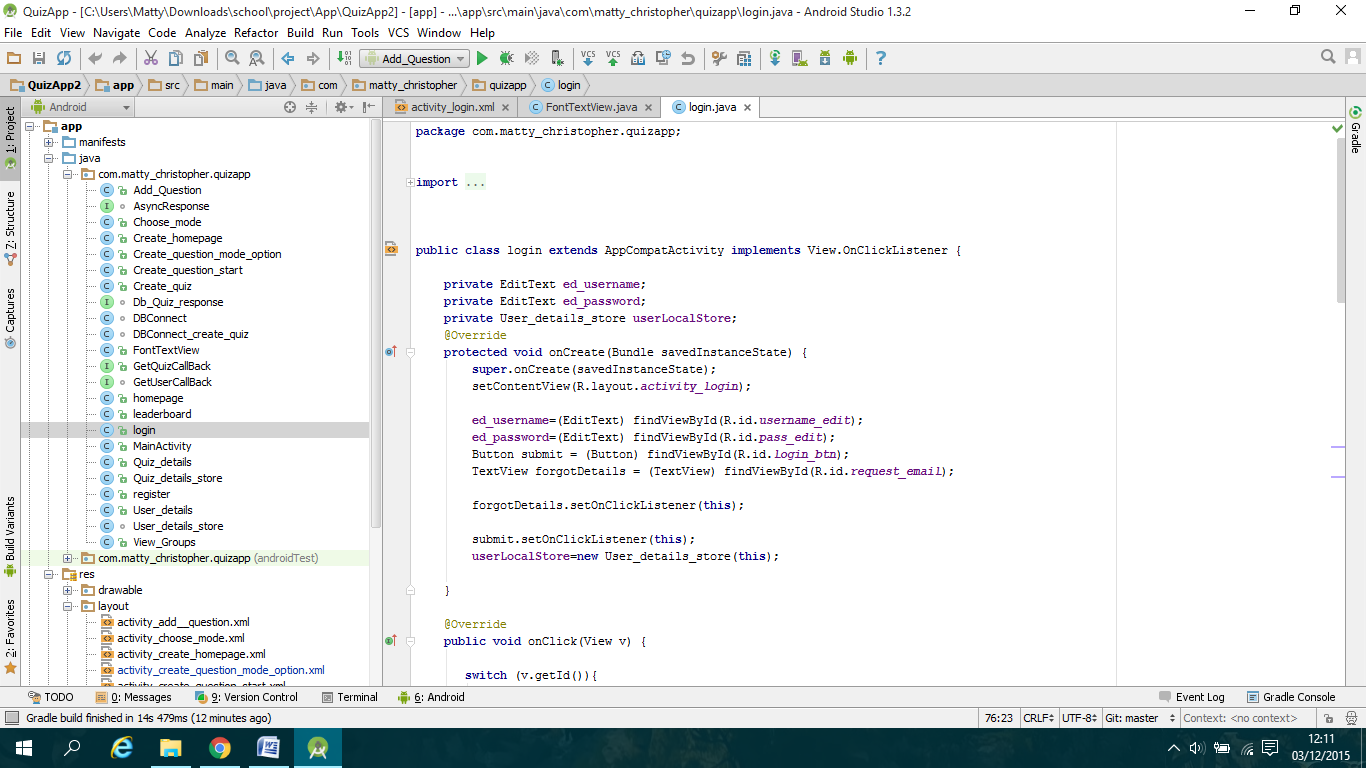
To create the interfaces I added components such as buttons and text views to the .xml files in the layout folder for each created activity. Here I used the Relative layout which allows me to display components on different parts of the screen easily without having to use multiple linear layouts.



To create the headings of each panel such as Login I decided to use the delarge.ttf font which mimics a marker pen font. To do this I used this class-which I have named FontTextView, below which I found online[reference]: http://www.101apps.co.za/index.php/articles/using-custom-fonts-in-your-android-apps.html



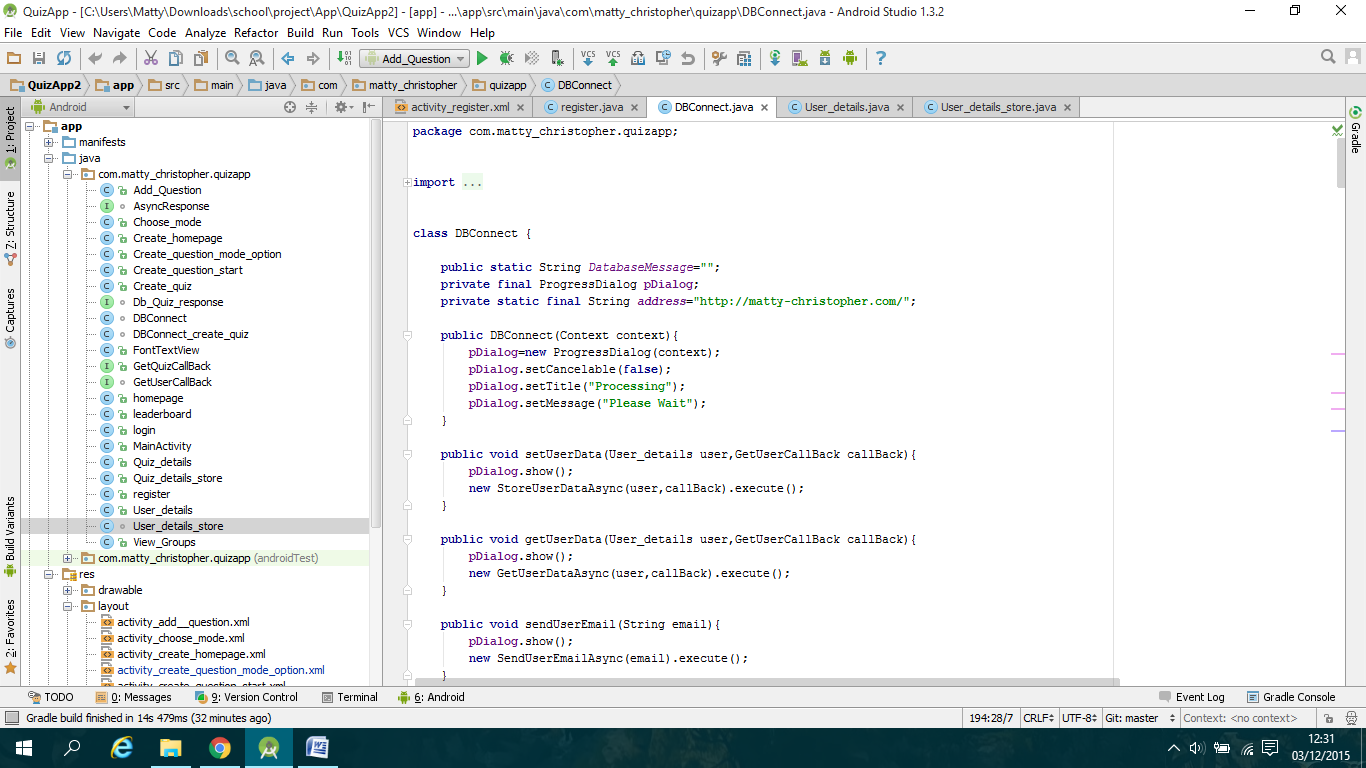
This class allows me to create a custom text view in the xml layout files so I don’t have to keep calling the createFromAssets() method in the java files for each panel.

Then to add functionality to components such as Edit Text and buttons, I created private variables inside the java file for the corresponding panel where I would cast the component on the xml file to the new variable so I can add an onclick listener or retrieve text entered into the textboxes: 

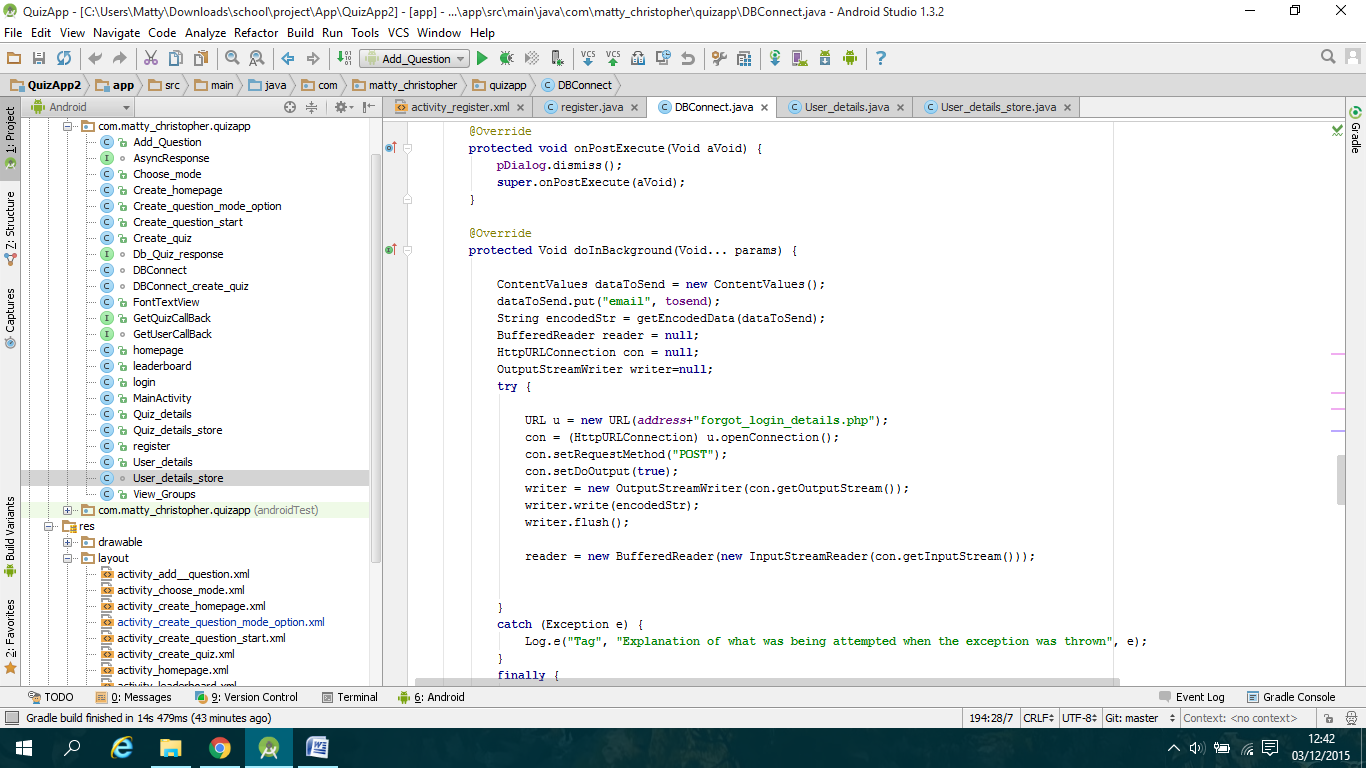
*Creating the database classes:*

As my app requires a database I needed to create a series of php scripts which are stored on my server at "InMotionHosting.com". I set up a "DBConnect " class which handles all of the scripts for logging in and registration of users. First I created a "User\_details "class which stores all the data belonging to a user such as username, password, email, etc.. I then setup a shared Preferences class called "User\_details\_store" which allows data from the "User\_details" class to be shared between classes.

The next step was to create a static variable to hold my domain address of where my scripts are held and a ProgressDialog variable to allow use of a progress bar to show the user that the data was being retrieved. To start the progress dialog I created a public constructor for "DBConnect" in which I initialised the dialog bar in the current panel using Context context. Then a series of methods are implemented which are used to display the dialog bar and execute the correct AsyncTask class:



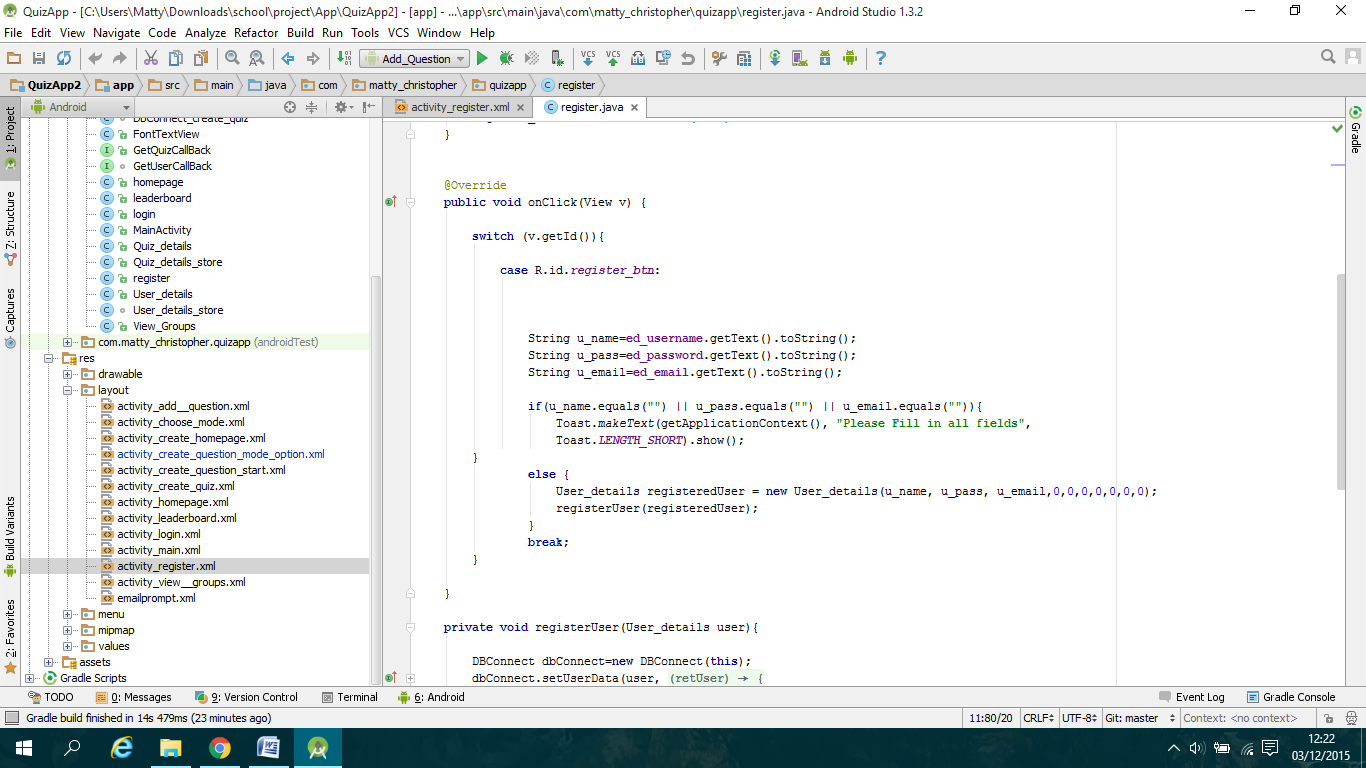
The next step was to create the AsyncTask classes such as "StoreUserDataAsync" which allows tasks to be executed on a separate thread so that they can be executed in the background therefore not blocking other threads. I used 2 of the 4 methods for each AsyncTask in the "DBConnect" class which are: "onPostExecute()" and "doInBackground()", the "onPostExecute()" method is called after the "doInBackground()" method has finished executing. A call-back method will be called which will allow data retrieved from the scripts to be used on the section classes:

**

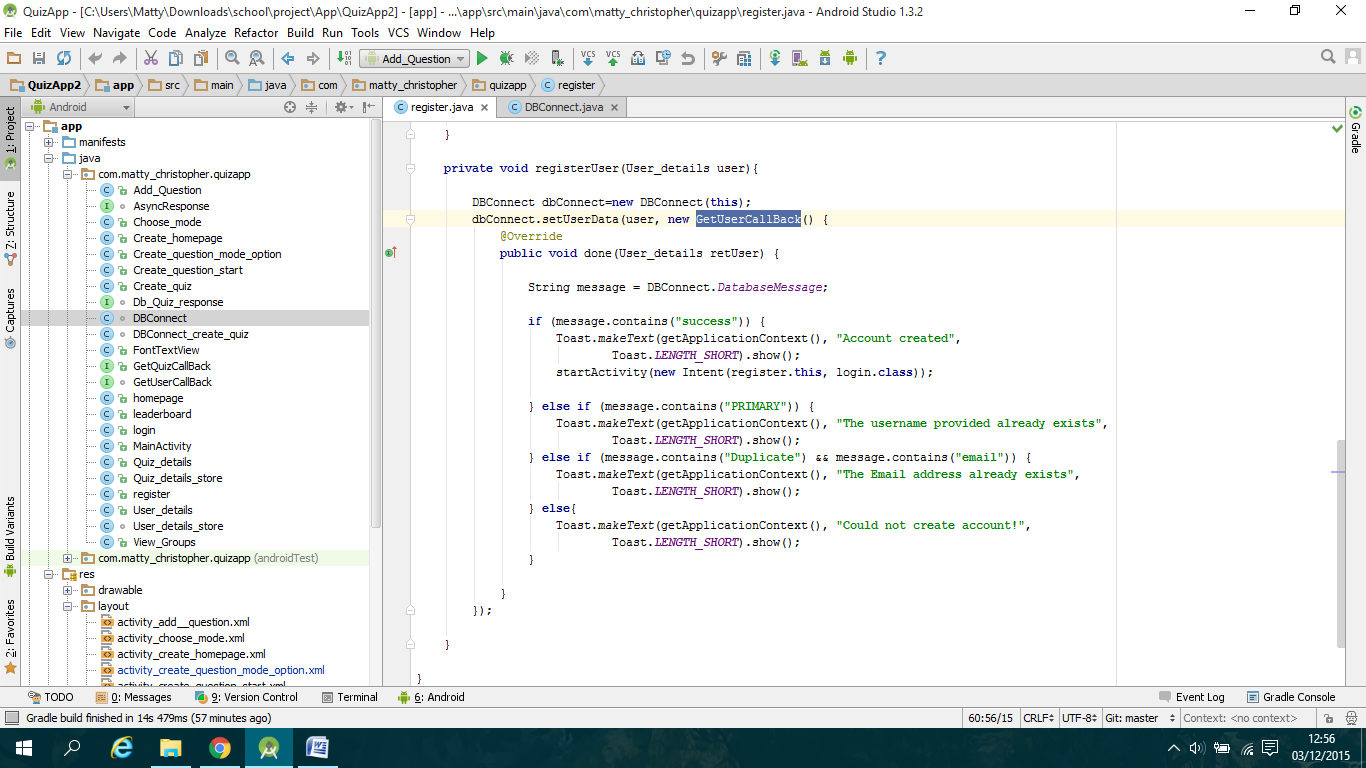
To make a connection I used the "HTTPURLConnection" class which allows sending data to the script and then getting data back from the response. Data to be sent is stored in ContentValues which gets data stored from the "User\_details" class.

*Creating the registration section:*

To start with I created the interface through the xml layout file called:"activity\_register.xml" and then giving the components functionality in "register.java". The next step was to create the onclick listener for the register button which I made the register class implement "View.OnClickListener" which let me implement the "onclick()" method. In this method I used the "getText()" methods to retrieve the text from the edit text fields which I stored in new local variables where I then was able to use an if else statement to check if any of these fields were empty, here a Toast message would be displayed if any field was empty:

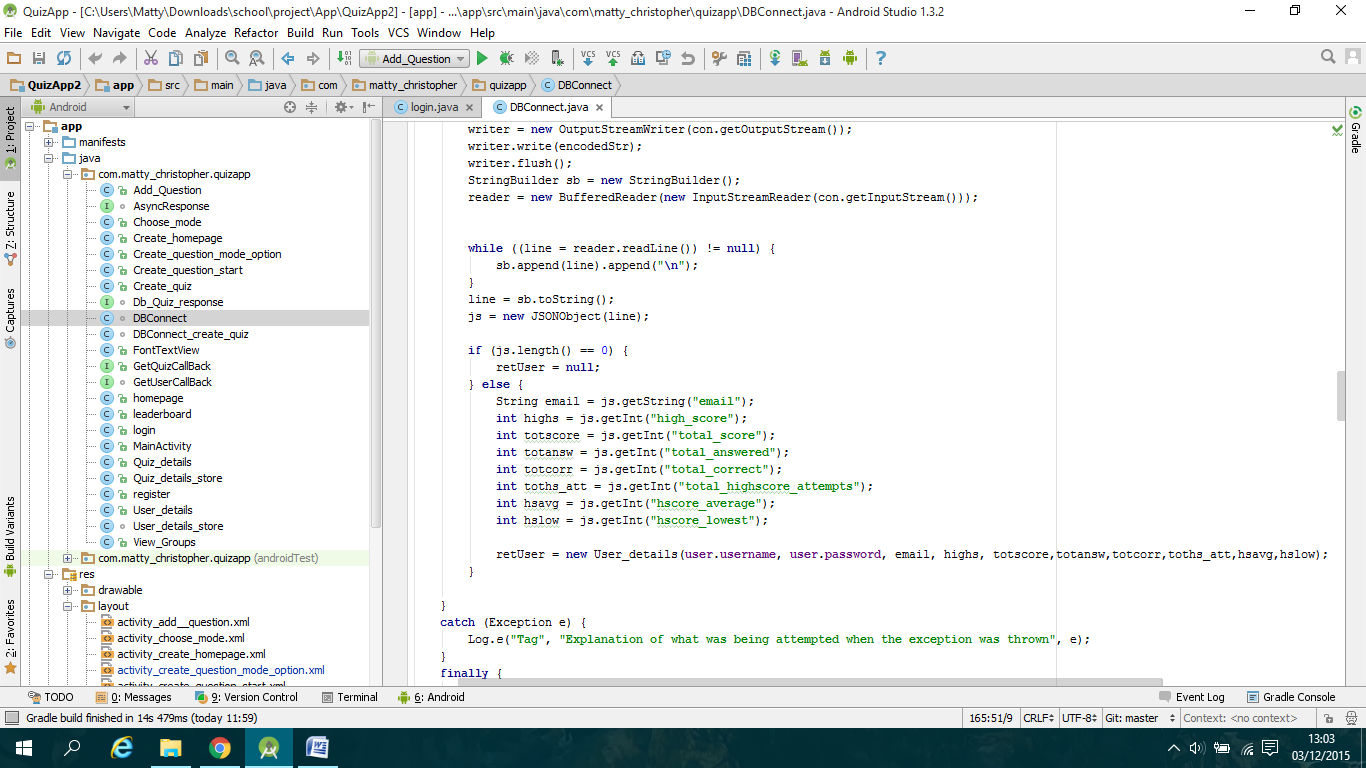


Otherwise the database class would be called. I created a new method "registerUser(user)" which handles calling the "DBConnect" class. A new object of the class is created which then allows me to call the "setUserData()" method which in turn executes the "StoreUserDataAsync" class which calls the "register.php" script on the server. I implemented a global variable to keep track of the return message of the script which I then call in the "done()" method from the passed through "getusercallback" class, here I implemented an if -else if- else block where I display a series of Toast messages depending on the response by implementing the statements to use contains() which look for specific words. If the response is "success" i call the "startActivity()" method to change the panel to login.



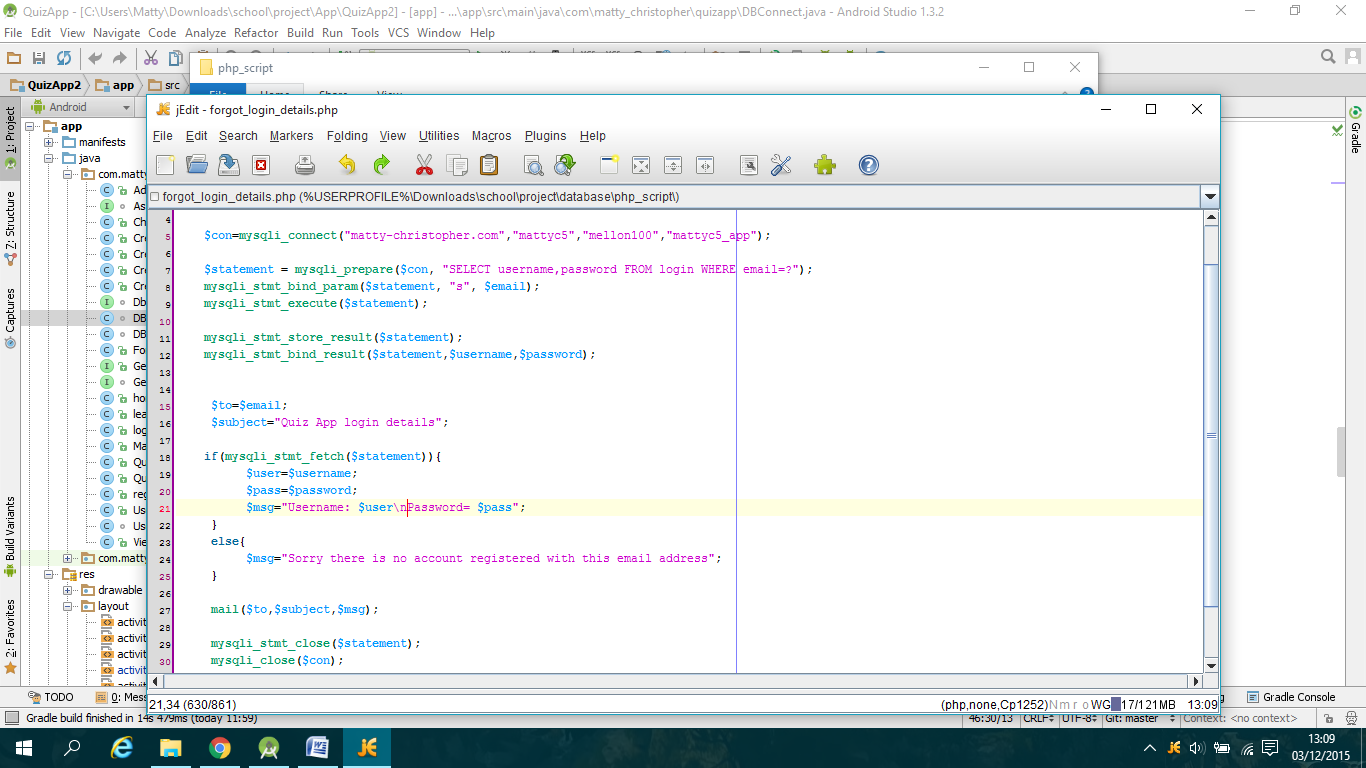
*Creating the login section:*

For the login class my implementation is similar to the register class for setting up the interface and calling the database. Here I call the "getUserData()" method in the "DBConnect" class which calls the "GetUserDataAsync" class which I implemented to call the "login.php" script. Here if the response is successful I implemented it so that the returned value Is the new logged in user which is then stored in the "User\_details" class as retUser object.



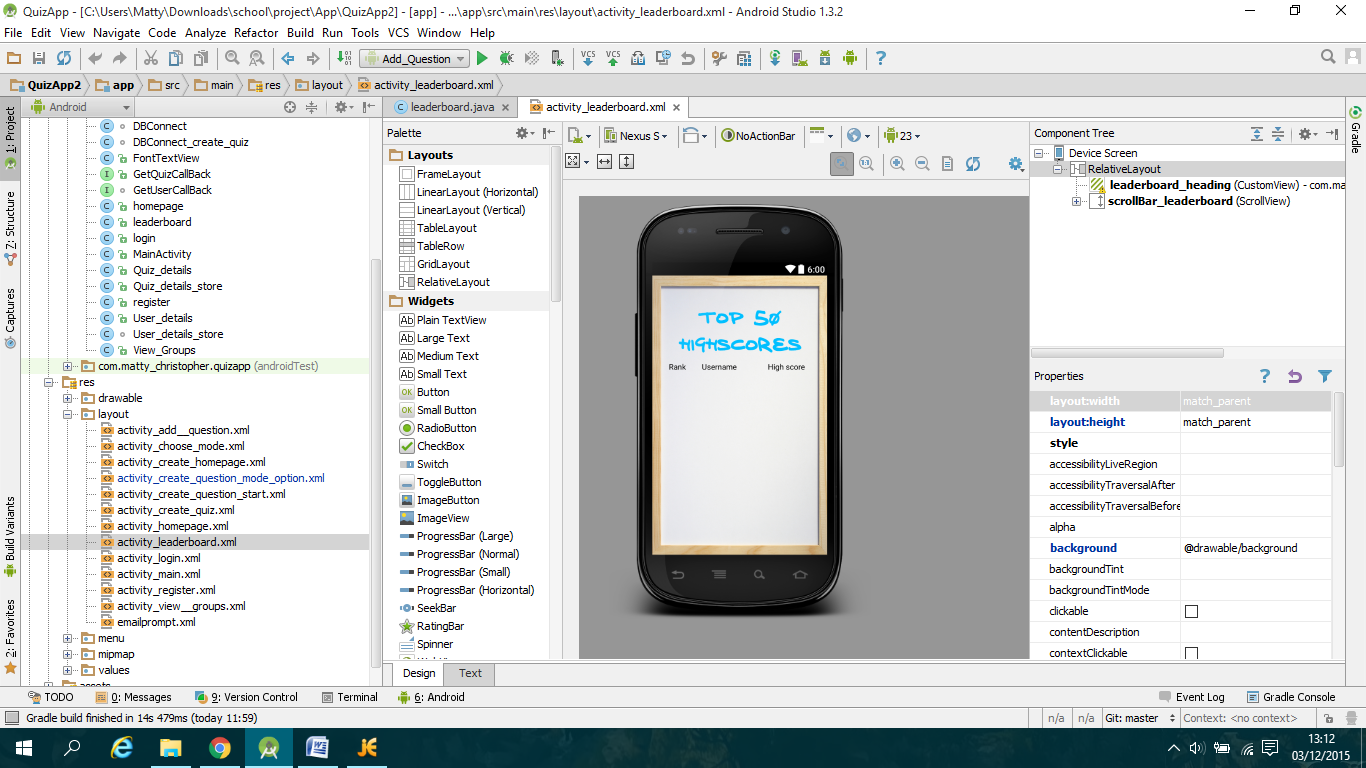
If the returned user is not null then the login class will call the "startActivity()" method and store the user details into the "User\_details\_store" shared preference class.

I also implemented a textbox to be a button in which allows users to send an email to request their details. To do this I created another "DBConnect" object which then calls the "getUserEmail()" method which will execute the AsyncTask class which sends the email address provided by the userto the email.php script which will send an email with the username and password if it exists or a message saying that this email address has no assigned user.

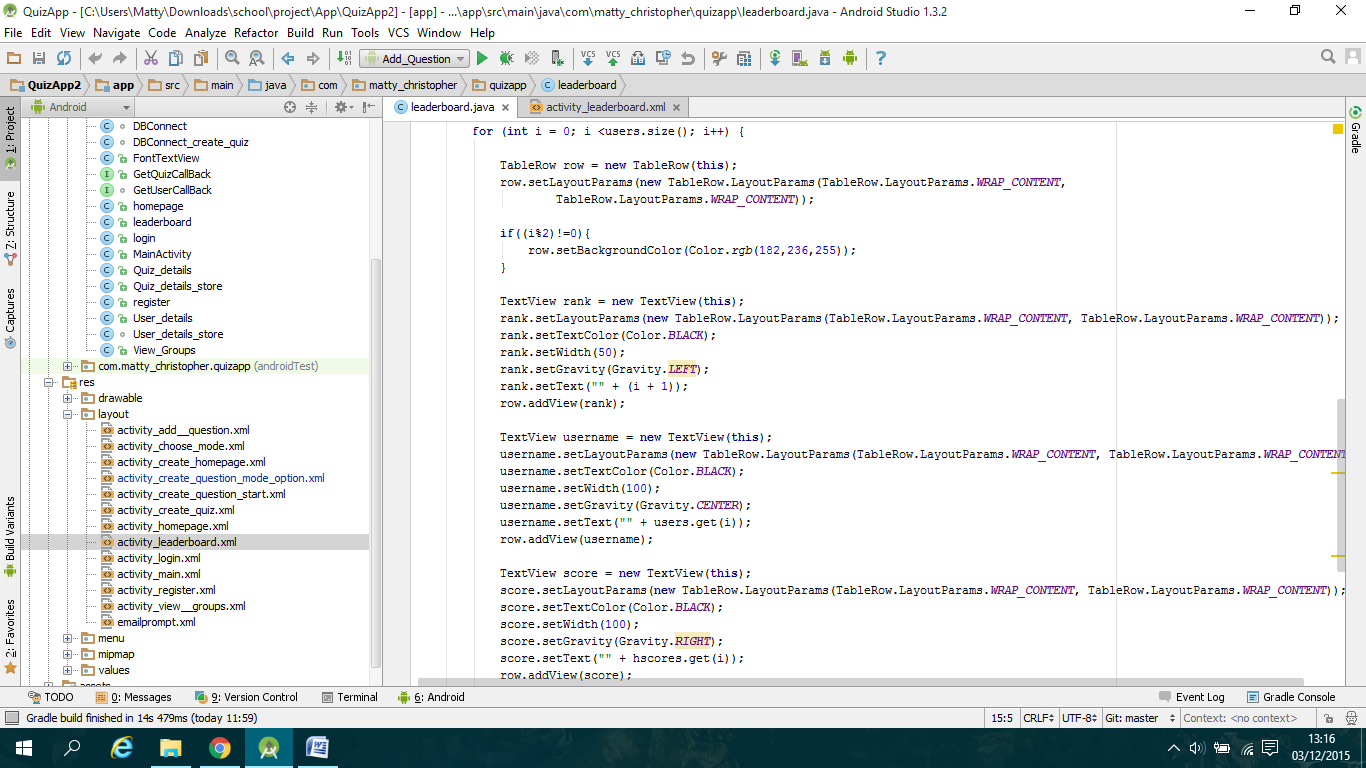


*Creating the leaderboard section:*

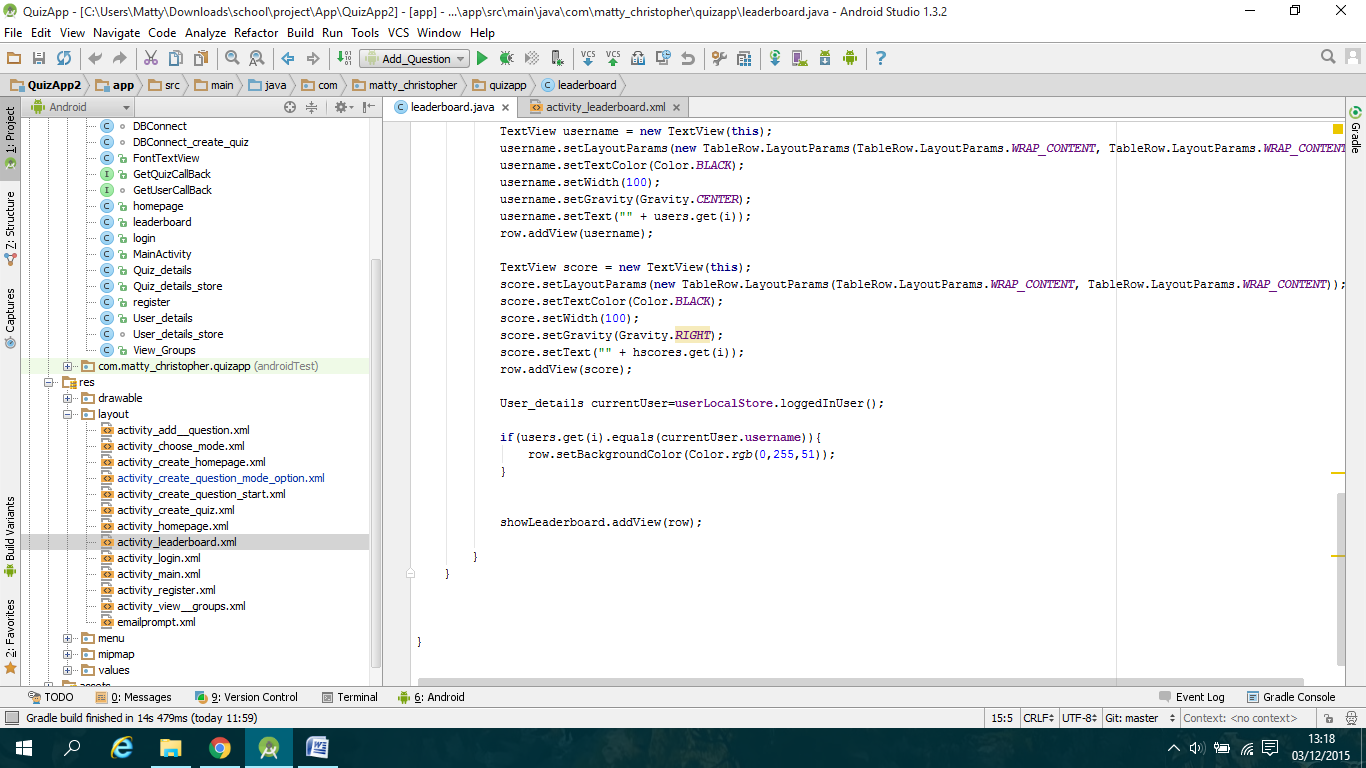
To create the leaderboard section I initially created the table component with a single row which displays rank, username and highscore text fields, I then added a scrollview ontop of the table to allow users to scroll up and down, this is what the screen initially looks like:



I then implemented the database connection which retrieves the data from the script and stores the data into 2 arraylists: user and highscore. Here I then created a loop which goes around user.size()-1 times which inside the loop I create new rows for each user.

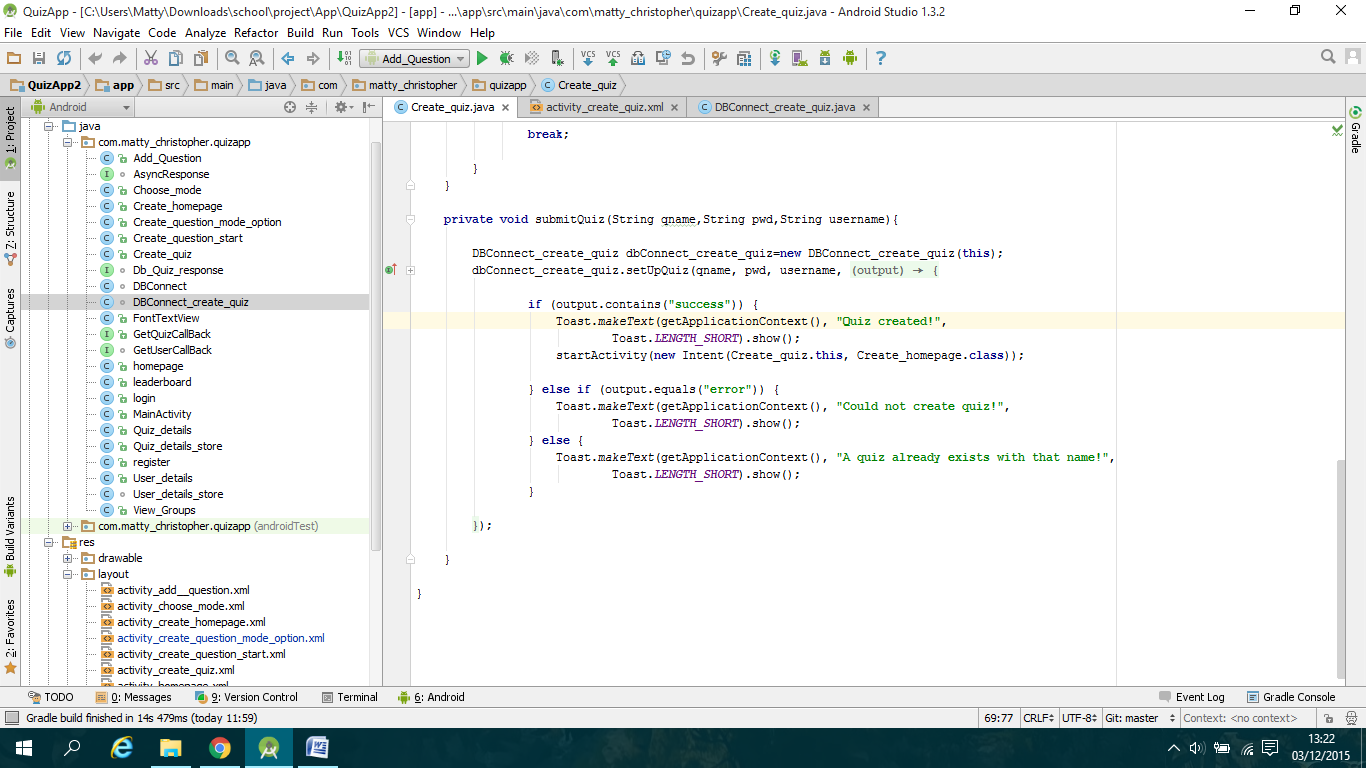


The final step was to implement an if else statement at the end of the loop which checks if the current user matches the user in the shared preference class, if the user matches then I made that row's background green so that the user can clearly see their rank.



*Creating the create quiz section:*

To create this section is very similar to login as there are two edit text boxes to enter a quiz name and password, here the "DBConnect\_create\_quiz" class will be called which calls the php script to check whether a quiz with that name exists or not, if it exists then a new entry is added with all the stats colums set to 0. Otherwise a toast message is called and displays an exists already message.



*How I will implement create question:*

To implement the create question section I have made the login to the quiz step where the user selects the quiz they want to add the app to. The next step is to add the panel which deals with choosing whether to create the question as a group or on their own. To create a question on their own the panel will display the logged in username for each answer and question, whereas with creating a group the user will enter 3 other usernames which will then be checked against the user table in the database to see whether they exist. Then on the question panel the user will have the option to assign each member to part of the question through drop down boxes next to each answer edit text box. When users edit the question the panel will check the shared preferences logged in user against the assigned text box, wherever they are matched they will be able to edit that text box.

*How I will implement view groups section:*

To implement this section I will create a table with a scrollbar attached to it similar to the leaderboard section. On each row returned will have an onclick listener which will allow the user to select a row and then be taken to an edit question panel where they can edit the assigned part of the question. To do the script I will have to send the current users username to the script which will have a query which checks the username against the groups table and will store each row where that user is part of into an array which will be returned. This array will then be used to fill each row in the table.

*How I will implement play section:*

I will need to create a new question class which stores the question along with the four answers and correct answer number. I will then need to create scripts which retrieve questions from the database and store them into the class. I will also need to keep track of stats when a user is playing by storing them into a local arraylist/s and upload them to the database through a script at the end of the play through. This will be better than continuously calling the database at each question as there will be only one call making it quicker.