Design examples

2021-22 Coursework specification

Select and apply appropriate technique(s) to model:

1. The structure and flow of the interface.

   Model the user interface, that is the web pages that the target audience using your web app and dashboard will see.

   The teaching materials focus on the use of wireframes for this. You may use other appropriate techniques.

   There are no marks for colour/artistic design or novel interface interactions. Focus on the structure and information flow.

   Consider whether you are designing for mobile or desktop use (this will depend on your target audience).

2. The application design

   Focus on the web application functionality rather than the dashboard. The dashboard (charts/visualisations) will be designed in COMP0034 coursework 1. You do not have to provide an object-oriented design, however there are common design patterns that may be useful to consider.

   You must use Flask to create the web app and Plotly Dash to create the dashboard in COMP0034. This will likely constrain your application design. The teaching materials discuss how use of these frameworks may affect your application design.

   You may use any models you wish and/or use more than one model for the design. For example, if you are using UML to model the application you might choose to provide a class diagram and one or more sequence diagrams.

3. The relational database design.

   Consider the conceptual and logical stages of database design. Design your database using a format that represent:

   - the tables (entities)
   - the attributes of those entities with data types and any constraints
   - relationships between tables

   An ERD or a table style schema are likely to be the most suitable format to model the database design.

   Your design is likely to have few tables since the web app is not complex. You do not need to include your dataset in the database as you already prepared this in coursework 1 (though you may include it if you wish).

Merit example

Tutor comment

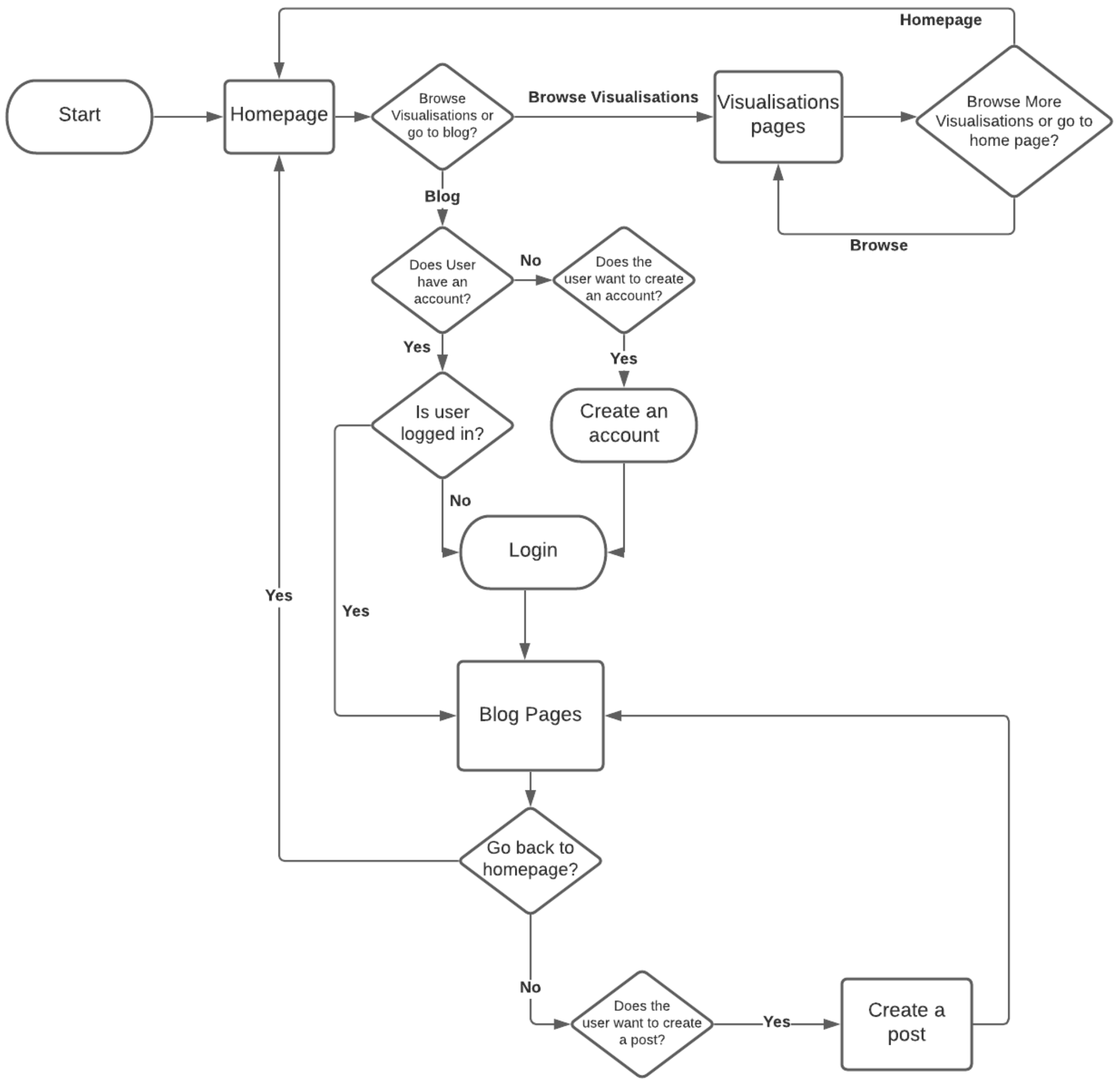
Feedback given to the student "The flow chart has been used to appropriately show the user journey through the web app. The wireframes are well presented with an appropriate level of detail. The routes and classes are appropriately presented. The ERD is well drawn though there are a couple of small issues. You don't need 'replies' in the Post table as you can find the replies the Post Reply table by using the linked post_id FK field. You probably don't need the original author id either, this is duplicate data as you already have that in the Post table."

Note that the flow chart is something this student researched and created and was not requested in the coursework.

Student submission

Structure and flow of the interface

A flow diagram is linked below which shows the pages and process for a user who is using the web app.

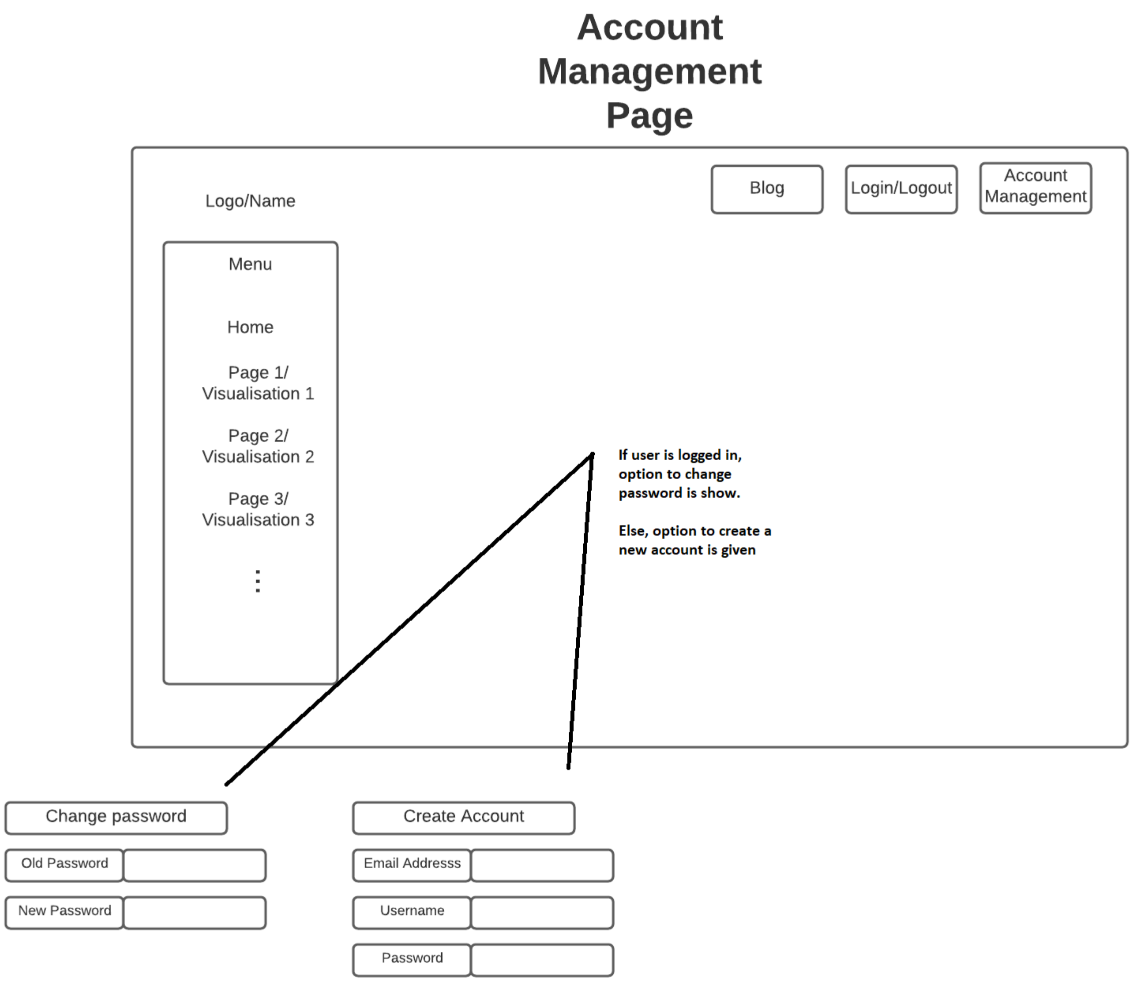


Start
Homepage
Browse Visualisations or go to blog?
Browse Visualisations
Visualisations pages
Browse More Visualisations or go to home page?
Homepage
Browse
Blog
Does User have an account?
No
Does the user want to create an account?
Yes
Create an account
Yes
Is user logged in?
No
Login
Yes
Yes
Blog Pages
Go back to homepage?
No
Does the user want to create a post?
Yes
Create a post

Each page of the web app has its own wireframe which is linked below. These show the user interface for the web app for each page and give an idea as to what the final version will look like.

(1) Homepage - the first page that is seen by the user on the web app
(2) Account_Management_Page - Where user can create, edit or delete account.

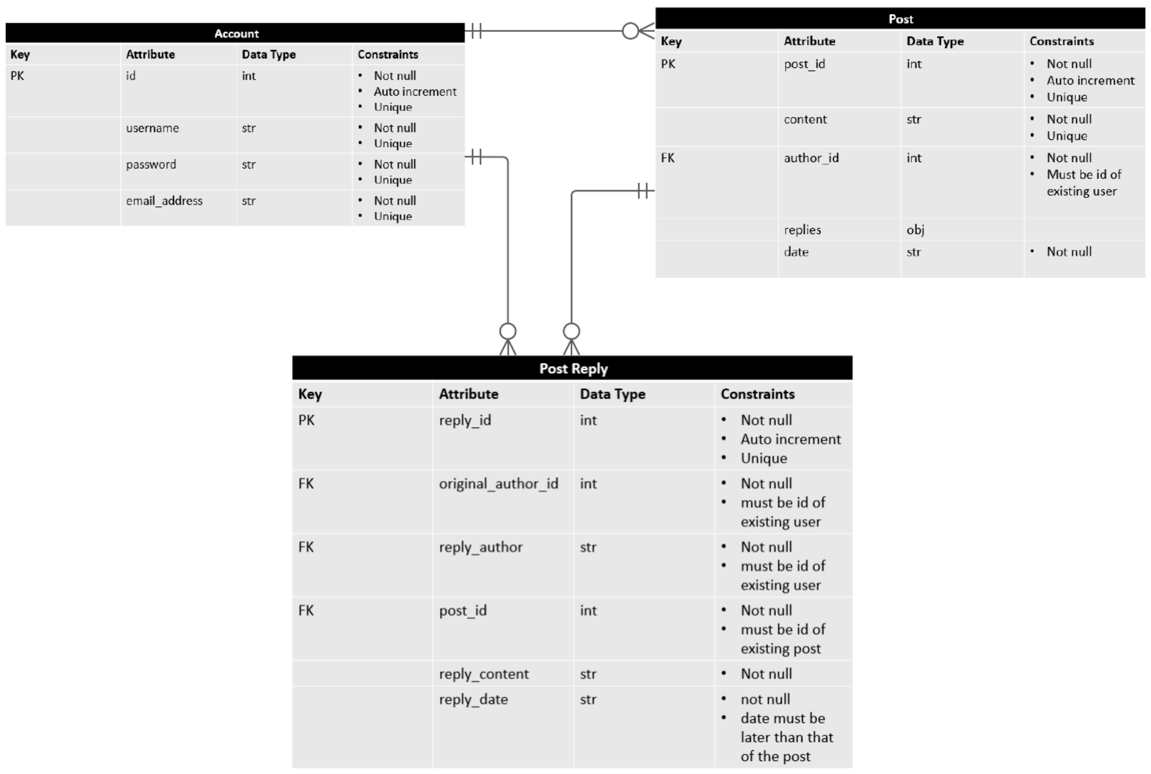
(3) Login_Page - where the user can log in
(4) Visualisation_Page - where the visualisations/ graphs on the web app are shown.
(5) Blog_Page - shows existing blog posts in reverse chronological order.
(6) Create_blog_post_page - Where the user can create a blog post.
(7) Reply_to_Post_page - where the user can reply to a blog post.



{The remainder of the wireframes have been omitted as the above should be sufficient to guide you as to the standard of work}

Application Design and Relational Database Design

The entity relationship diagram (ERD) is linked below for this database and includes the attributes of each entity as well as the relationships between entities. Each attribute also has some constraints.



Account

Key	Attribute	Data Type	Constraints
PK	id	int	• Not null • Auto increment • Unique
	username	str	• Not null • Unique
	password	str	• Not null • Unique
	email_address	str	• Not null • Unique

Post

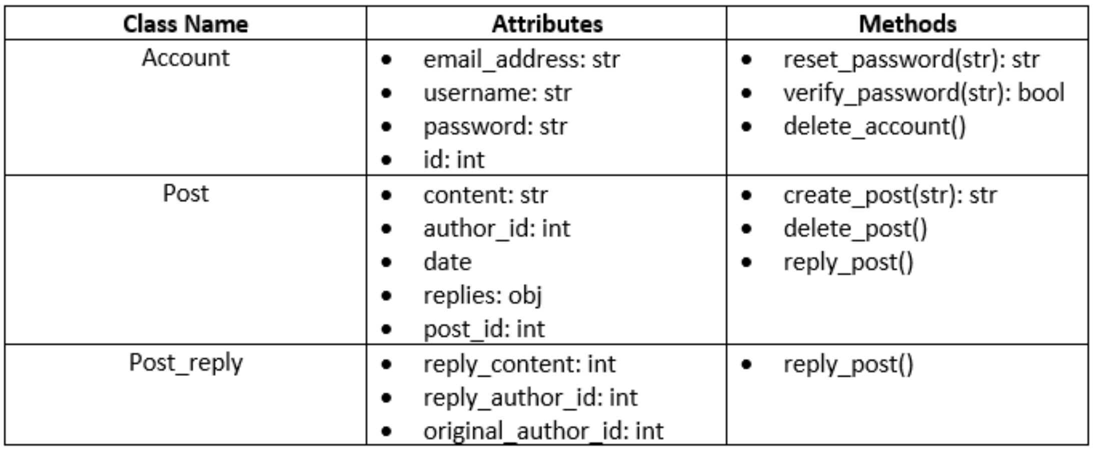
Key	Attribute	Data Type	Constraints
PK	post_id	int	• Not null • Auto increment • Unique
	content	str	• Not null • Unique
FK	author_id	int	• Not null • Must be id of existing user
	replies	obj	
	date	str	• Not null

Post Reply

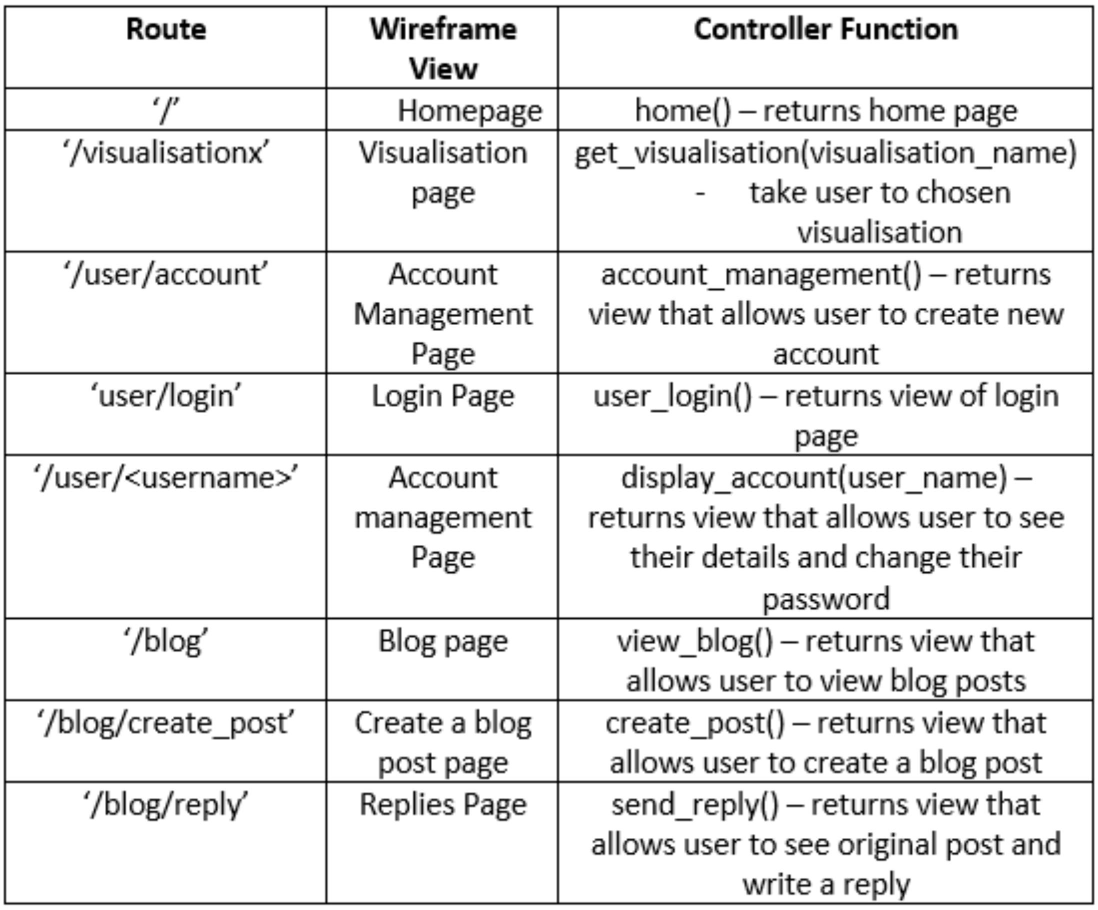
Key	Attribute	Data Type	Constraints
PK	reply_id	int	• Not null • Auto increment • Unique
FK	original_author_id	int	• Not null • must be id of existing user
FK	reply_author	str	• Not null • must be id of existing user
FK	post_id	int	• Not null • must be id of existing post
	reply_content	str	• Not null
	reply_date	str	• not null • date must be later than that of the post

Classes and routes

The classes used are those associated with the user's accounts and blog posts within the web app, these show the class names, the attributes as well as data types and some example methods which may be used. The routes are also shown as demonstrated by the MVC format (Model (classes), View (wireframes) and Controller (routes)), these show the URL, the appropriate wireframe for the controller and a description as to what the controller function is.



Class Name	Attributes	Methods
Account	• email_address: str • username: str • password: str • id: int	• reset_password(str): str • verify_password(str): bool • delete_account()
Post	• content: str • author_id: int • date • replies: obj • post_id: int	• create_post(str): str • delete_post() • reply_post()
Post_reply	• reply_content: int • reply_author_id: int • original_author_id: int	• reply_post()



Route	Wireframe View	Controller Function
'/'	Homepage	home() – returns home page
'/visualisationx'	Visualisation page	get_visualisation(visualisation_name) - take user to chosen visualisation
'/user/account'	Account Management Page	account_management() – returns view that allows user to create new account
'user/login'	Login Page	user_login() – returns view of login page
'/user/<username>'	Account management Page	display_account(user_name) – returns view that allows user to see their details and change their password
'/blog'	Blog page	view_blog() – returns view that allows user to view blog posts
'/blog/create_post'	Create a blog post page	create_post() – returns view that allows user to create a blog post
'/blog/reply'	Replies Page	send_reply() – returns view that allows user to see original post and write a reply

Pass example

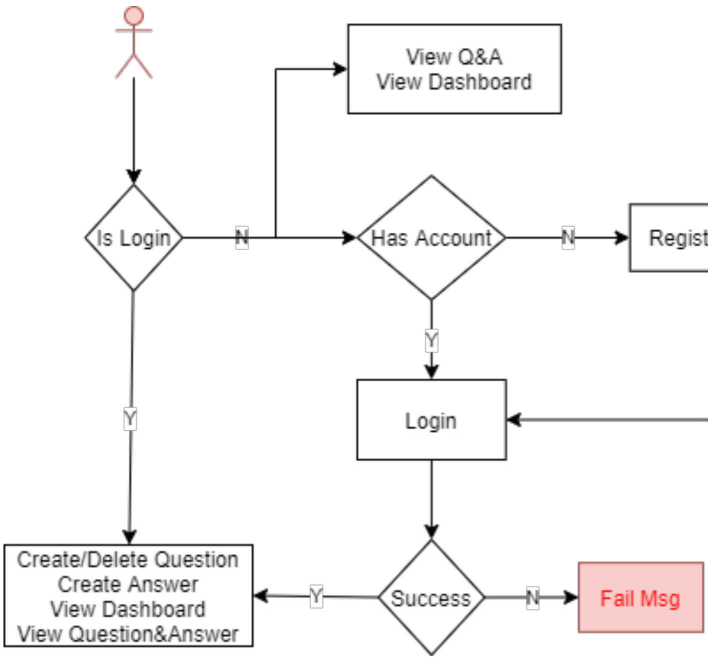
Tutor comment

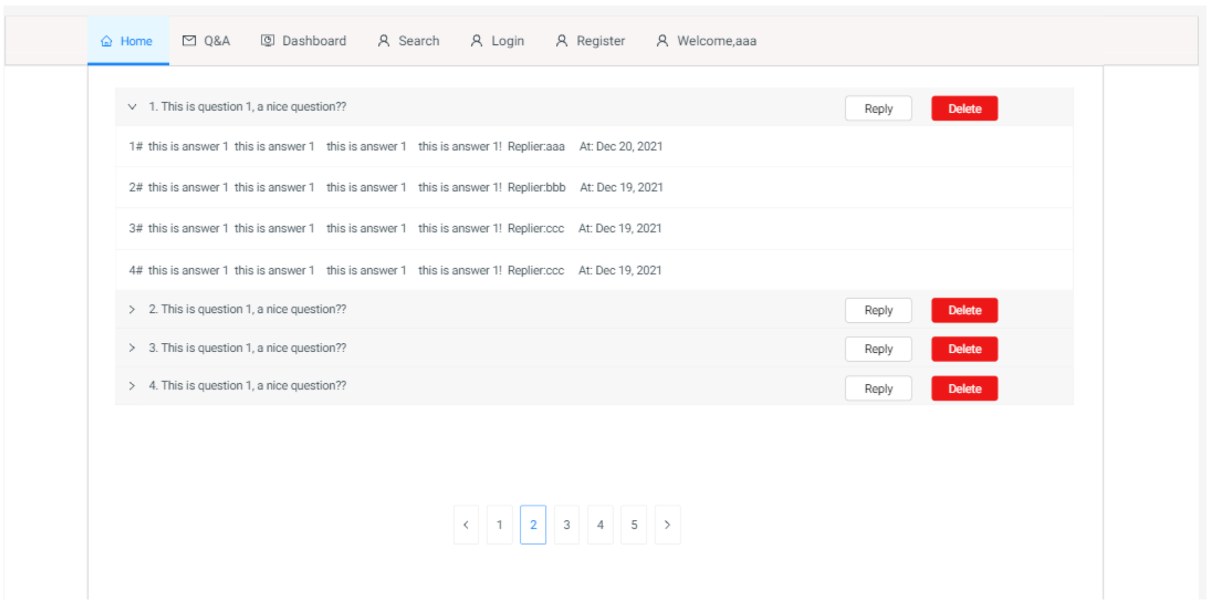
Feedback given to the student: "The wireframes are clear and relate to the user stories, though they don't cover all of the functionality of the app. In contrast the routes appear to cover the requirements. You haven't provided the classes which is a large part of the application design. The database design is mostly appropriate though you don't use SQLite data types (varchar and datetime don't exist in SQLite) and the constraint detail is missing. If you have a relationship between tables then the primary key from the parent should be in the child table and labelled as the foreign key. In your diagram the id from the user_info should also be in the answer table and question table and isn't.”

This student missed an element of the coursework which led to a lower mark; otherwise the standard of their work was of a merit level.

Student submission

Structure and flow of the interface





1. This is question 1, a nice question?? Reply Delete

1# this is answer 1 this is answer 1 this is answer 1 this is answer 1! Replier:aaa At: Dec 20, 2021

2# this is answer 1 this is answer 1 this is answer 1 this is answer 1! Replier:bbb At: Dec 19, 2021

3# this is answer 1 this is answer 1 this is answer 1 this is answer 1! Replier:ccc At: Dec 19, 2021

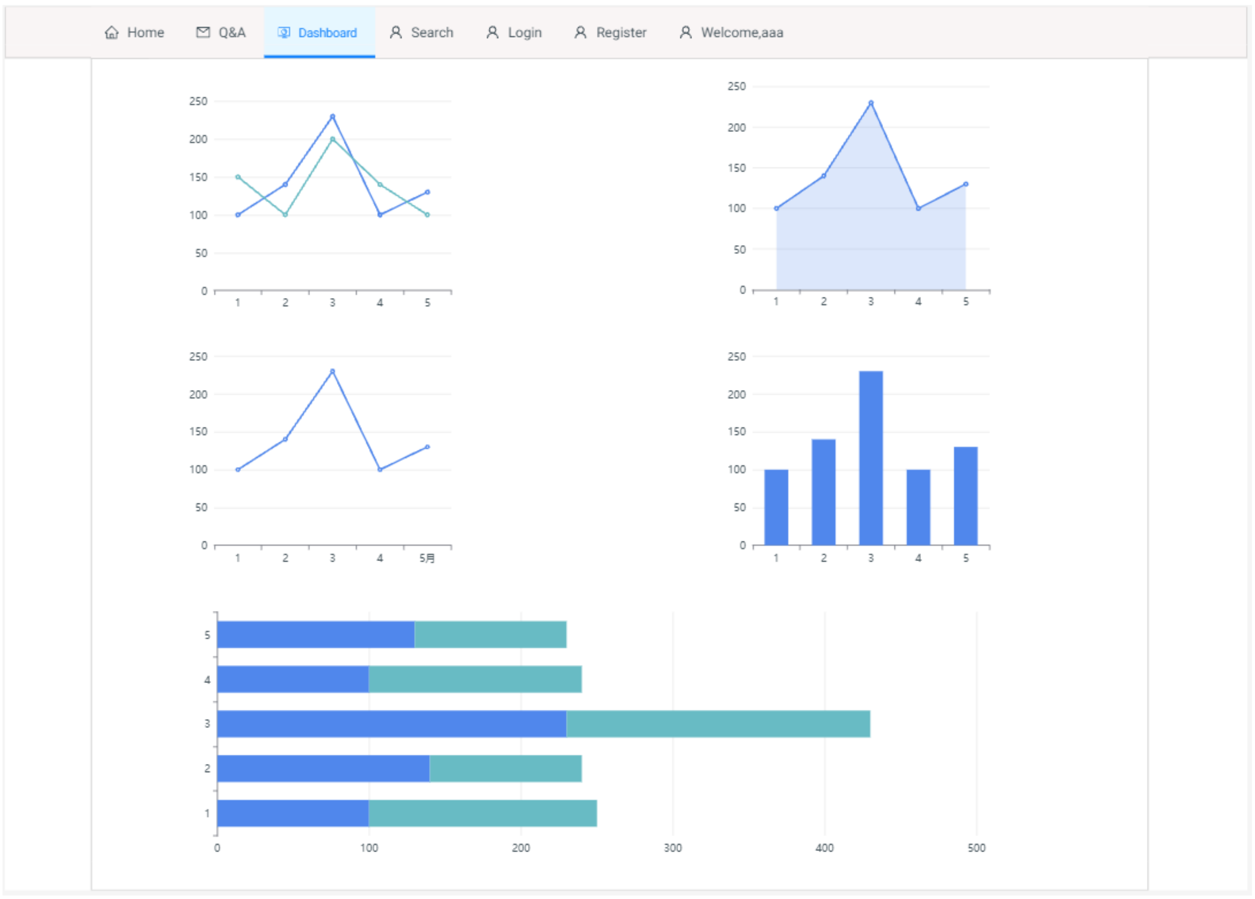
4# this is answer 1 this is answer 1 this is answer 1 this is answer 1! Replier:ccc At: Dec 19, 2021

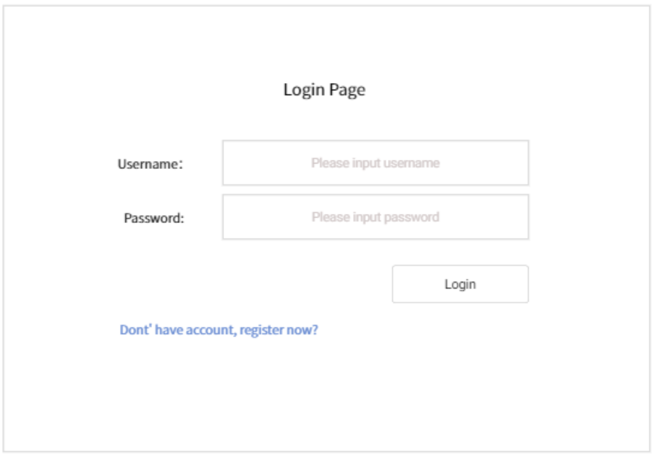
2. This is question 1, a nice question?? Reply Delete

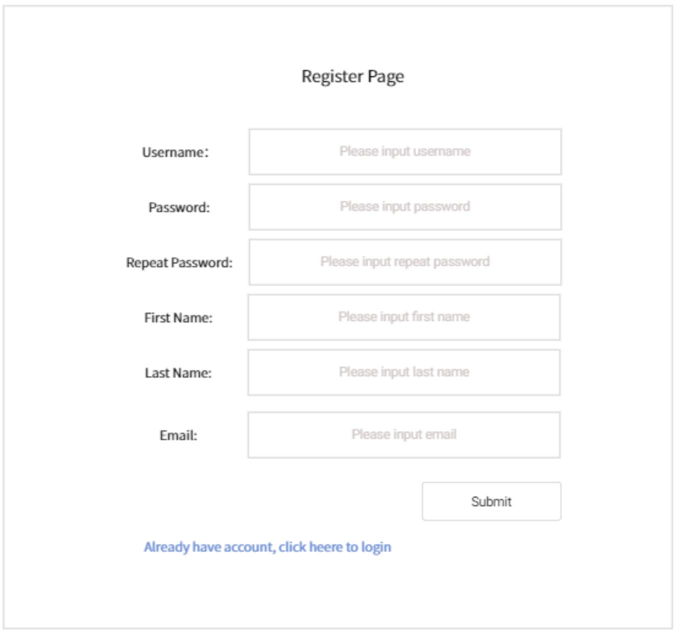
3. This is question 1, a nice question?? Reply Delete

4. This is question 1, a nice question?? Reply Delete

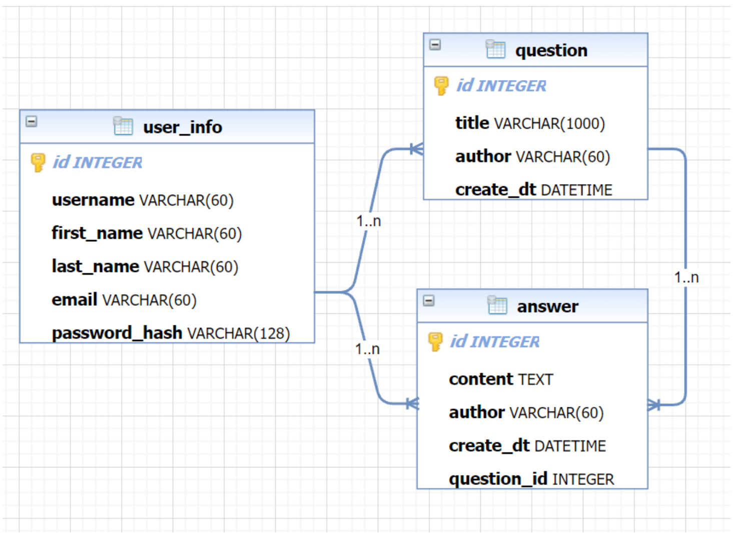
‹ 1 2 3 4 5 ›


Home
Q&A
Dashboard
Search
Login
Register
Welcome,aaa
250
200
150
100
50
0
1
2
3
4
5
5月
100
200
300
400
500





Relational database design



Application structure

Route	View (wireframe)	Controller function
'/welcome'	Welcome page	index () Renders welcome page
'/login'	Login page	login () Takes the entered account info, checks against the details in the database, returns error if details incorrect otherwise redirects to users account page
'/register'	Register page	go_to_register () go to register page
'/login/don't have account'	Register page	go_to_register page () go to register page
'/register/already have account'	Login page	go_to_login page () go to login page
/login/login'	Login page	login () Takes the entered account info, checks against the details in the database, returns error if details incorrect otherwise redirects to users account page
'/register/submit'	Register page	Submit () check password and repeat password same or not, return try again If not.
'/home'	Home page	go_to_home () go to the home page
'/dashboard'	Dashboard page	plot () read the data and plot the graph
'/Q&A'	Q&A and homepage	go_to_Q&A page () go to Q&A page
'/home/reply'	Home page	Reply () add text on the answer and show the username and time for replying
'/home/delete'	Home page	Delete () delete the question
'/search'	Home page	Search () check if the str matches the data set in the database, returns error if details incorrect otherwise redirects to data set being search