The Team BELL Learning App Project Report

CFG Software 3 cohort 2022
Team members: Esther Makinde, Linsay O'Hare, Rebecca Sewell and Louise Anderson

INTRODUCTION

The aim of this project is to design and produce a revision application as part of the CFG Software Nanodegree course requirements. The course itself is very intensive, and we felt a revision application would provide an easy to use, focused area of study for users already balancing the commitments of full-time work, caring and/or family responsibilities.

BACKGROUND

We intend for the app to be used for CFG students to help revise the languages that they come across during the foundation part of the course. The application was created in the context of simultaneous/competing demands on the project team's time in the sense of the project running parallel to day jobs, the software nanodegree final assessment and weekly homework. If we were full-time developers we could assume this would be the focus of the company until completion and delivery.

SPECIFICATIONS AND DESIGN To make an app accessible to a larger range of users it is vital to check not only the code but the design and flow of the app. During the design phase, it was important to take into account colour accessibility for users with a range of sight issues which could exclude them. The two main colour contrasts of the page scored 9.06 in contrast which makes it suitable to use for both small and large text. The colours are #182E4B and #66E8C1 which not only tests high in contrast but shares a cooling, calming blue tone suitable for quiz taking as it helps reduce stress compared to warmer and aggressive colours such as red. It was also designed with a well spaced layout in mind so as not to clutter the screen and overwhelm the user during their quiz taking, keeping focus on the questions and allowing each answer to be clearly defined so as not to accidentally press the wrong one.

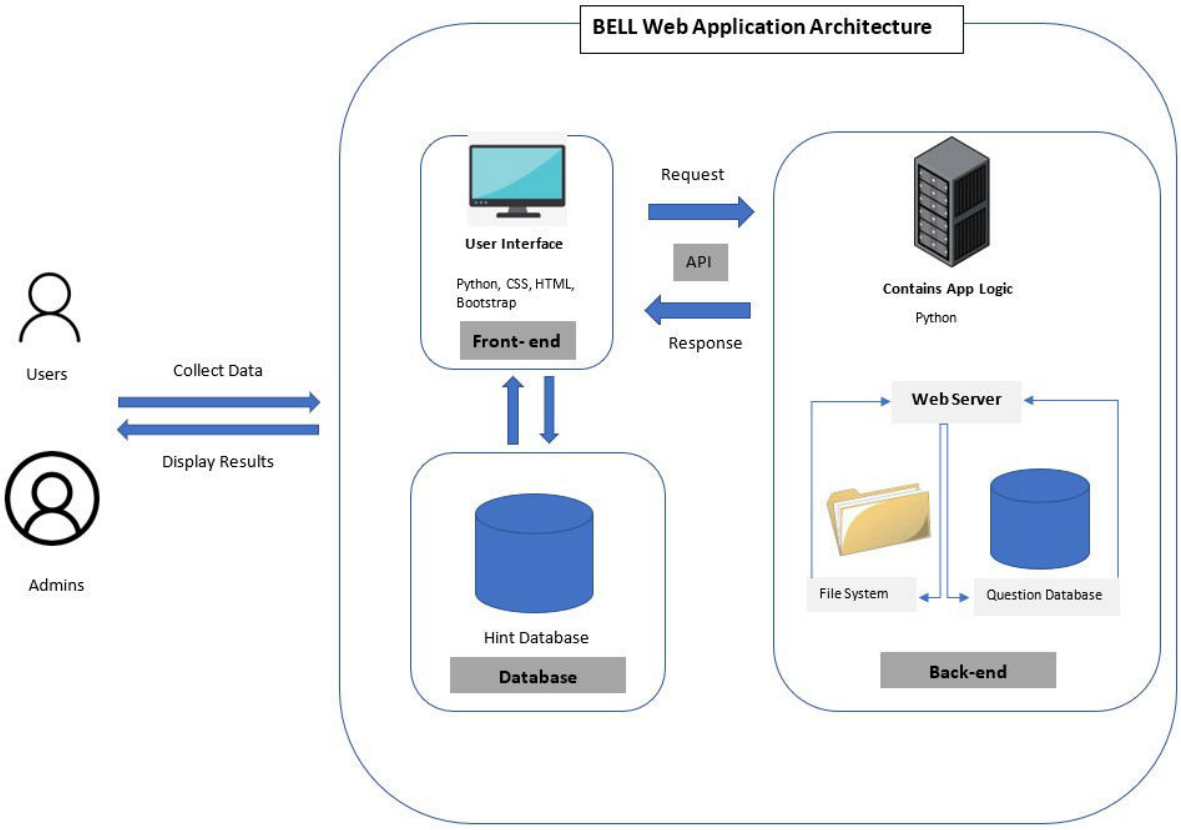

Figure 1 - Architecture diagram displaying the interaction between the various components of the BELL revision app.

IMPLEMENTATION AND EXECUTION

Development approach and team member roles:

To develop this app as a team an organisational list was created. Using the Notion Task board we generated a list of tasks and broke them down into smaller sections then listed them on the task board. Within the task board the individual items were given a priority rating, a dropdown menu that allowed its completions status to be updated—this also allowed to see if anyone was working on something already if we couldn't contact each other—by each developer, a short description of what was needed for each section, and a name tag so jobs could be assigned to each person.

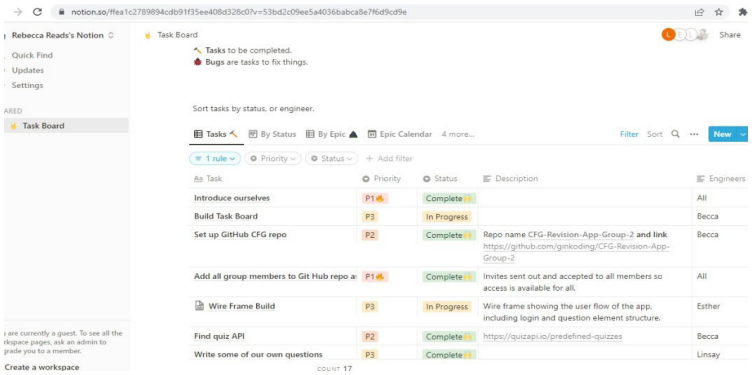

Figure 2 - image of Group Notion Board

This was also useful for sharing and storing websites/notes/articles that were discussed during video chats and messenger discussions. Using GitHub as version control also allowed each person to share their code and files within the group as they were being worked on and gave a level of security to merging codes and files together.

Tools and libraries
Flask
SQLAlchemy to connect to the DB
Python

Agile development Agile methods:

- Kanban - To Do, Doing, Done (Notion set up)
- Elements of Scrum
    - Weekly Sprints - Determining the features that we want to have developed during weekly meetings.
    - Hard to fully implement Scrum as we are just the developing team. Would have been easier to do with a Scrum Master and Product Owner.

Implementation challenges
Git proved to be initially challenging. Another challenge was moderating our ideas to match our current skill sets - our weekly meetings were very inspirational, however it was often a complex and ongoing task to marry up idea and successful execution. The timing of the final project was not ideal in the sense it was due in the same window as the heavily-weighted software final assessment. There was also a freakish heatwave so keeping our respective desktops/laptops and workspace environments was a novel problem. Fortunately the Autumn CFG cohort will not have the same issue.

Post project completion: The MVP of any project is to make sure the bare bones are up and functioning correctly. Without this there is absolutely nothing else to focus on in regards to working on the code, you need a foundation to build on.

In regards to the Team Bell Revision App, this means that our basic app will have several basic areas—login page, user account page, quiz start page, question page, results pages and logout page. The login page will be operational, with the ability for a user to update their details, only the most basic required—name, user name, password. The quiz will be pulling in from the https://quizapi.io/predefined-quizzes API and presenting the data as multiple choice to minimise user input error. The results page will channel the results of the quiz and present them as integers ex. 9 out 10 correct as opposed to percentage. Second to the basic structure being created and functioning is the general updates that will come

with any project. Continual upgrades to servers and databases will be needed to keep code operating with minimal interruptions to any additional branches of data and the operating of a site by a user. Once these are accounted for we would follow a basic deployment pattern, which would be repeated for following branches of content.

This list does not include planning, designing and designating tasks, just code building steps up to live release.

1. Step one - writing the code. this will have been determined and broken up in a scrum style.

2. Step two - once the basic code is laid out and functional it will be peer tested for snags, errors and bugs. This will be solved within the dev team before moving onto the next stage.

3.Step three - the code will be deployed to a development site and shared with shareholders and relevant parties before it goes live. From feedback at this stage, reverting back to stage one and through stage two may be necessary.

4.Step four - once shareholders and developers are satisfied with a functioning app it can be made live for public use. This comes with testing live on release for immediate fixes. This can involve listening to user feedback before deciding which area to add to the app for a better user experience.

TESTING AND EVALUATION

Evaluation/Limitations of the system

- Web usage mining
- User classification manager - admin or normal user
- Review the time-space complexity of snippets of code?
- Evaluate how easy it is to build on the current system
- Would we want a subscription model for future versions?

A measurable, reasonable goal of success for the project would be a user achieving an increase in correct answers over time and becoming more confident in their software journey. We believe the app fits in well alongside other course-created study aid applications and are fortunate to have a range of skill sets within the group. Components that could be reused for future projects from this app include the question database, log in code and design templates. The framework and code can be easily adapted to other subjects.

In order to minimise user error before release certain steps will be followed to recreate, test and correct any problems within the code before a real user can try. This involves several levels of testing, to catch any snags before development continues with an unresolved issue that can affect the functionality later on. One of the methods we used was peer testing, we shared our code on the version control system GitHub which allowed each of us to see the progress of the others and offer feedback at each step. Within GitHub, we had a Main Branch, and Developing branch, then when we created our own parts we created branches with the naming system Name/BranchNumber. Using this method we could keep track of our commits and easily identify who had created the branch. In keeping a Developing branch separate from the Main allowed us to have an extra layer of security when merging branches. To ensure no conflicts erased or created errors in code as we merged Name/BranchNumbers we merging into development and when they merged then tested successfully, we would move them from Developing to Main for presentation.

Checking for errors: Testing the functionality of the app for ease of use. For example, making sure all of the buttons are clickable and that they have the function they are labelled to have, such as a ‘Back’ button taking the user back to a previous page rather than submitting the answer to a question. Syntactical errors in the text on the site which can make the navigation and use confusing or downright impossible for the user to interact with the page. For example- <aside> 💡 What does HTML stand for? </aside> 1. It doesn’t. 2. A tree 3. Balloons In this case the question has no correct answer, so the user cannot proceed. This does not just apply to the questions but all of the text on the page being presented. Errors in calculation may occur in producing the results of the quiz for a user. Since we are wanting to provide a result out of 10 at the end of each quiz round, the maths will need to follow good logic and coding errors will need to be checked before use.

New areas to develop outside of the basic app for the Bell Revision App. Each of these would be complex additions, not to be added all at once but worked on, tested and deployed over time as the app grows. Use of new API’s. Pulling in a variety of quizzes from the same overall theme but different topics to allow the user to have greater choice of topic and appeal to a wider range of interests and skills.

Comparison points. The ability to come higher on a table than other users to increase the need to play competitively.

Creating a leaderboard would allow comparisons and user motivation without being necessary to daily use. The leader board would not be fixed, but refresh on a weekly basis to make sure new users have a chance to aim for higher positions.

Share function. All great apps have share functions built into them to increase awareness of the app on peoples socials accounts and to gather interest in what they are. An interesting and rewarding share option would garner the interest of new users at minimal cost. The app should eventually offer to share results of quizzes to social media sites. They would also be able to share results from the leaderboard.

Goals. Awards given to a user when a certain target has been met. ideally completing 10/20/30 etc. quizzes with full marks. This would be awarded to the user as a badge for them to keep on their profile.

Calendar of use. A calendar grid showing the users visits and level of interaction over a month using a coloured square that gets brighter the more the user has interacted with a page. This can also tie into awards, streaks of 7 days/30 days to receive another badge to go on their profile.

CONCLUSION

In conclusion the team has created a viable revision app which we believe will be of help to users beginning their software journey. We have worked as a team delivering the majority of what we set out to do in a limited time frame.