Department of Computer Science

CS353[A] — Team Project (2024-25:Semester 1)

Project Report

Team Name: Team 14

Student Name: Songyan Lai

Student ID: 24250371

Student Email: SONGYAN.LAI.2024@mumail.ie

Project Name: CS Connect (CS Social Website)

Member who supplied the screencast and project repository:

Danila Gorelov (Danila.gorelov.2023@mumail.ie)

Team number	Student ID	Student Email
Kian Curran Mahon	22367856	Kian.curranmahon.2023@mumail.ie
Dawid Jan Stal	22746739	DAWID.STAL.2023@mumail.ie
Killian Patrick Maxwell	22388316	Killian.maxwell.2023@mumail.ie
Emmanuel Gyamfi	21327303	Emmanuel.gyamfi.2022@mumail.ie
Danila Gorelov	22425236	Danila.gorelov.2023@mumail.ie
Songyan Lai	24250371	SONGYAN.LAI.2024@mumail.ie
Brian Ezeanya	22303966	Brian.ezeanya.2023@mumail.ie

Table of Contents

Chapter 0 : Introduction to Agile Development (SCRUM):

Agile development has in recent times become a main approach in modern software introduction and growth. Over the years it has grown rapidly using a lot older and inflexible methods like, for example, the Waterfall model method, where the development happened in strict and precise one-way stages—requirements, design, coding, testing, and maintenance (Balaji & Murugaiyan, 2012). These methods didn't allow a lot of flexibility, and by the time the software was finished, it may no longer fit in the customer's wants and needs. Even though Agile development recently hasn't been up to standards with being flexible, when it was known to be flexible, it mainly focused on a lot smaller improvements to a person's Software Project (Schwaber & Sutherland, 2020). It highly supports teamwork, solid and honest feedback and can be prepared for any changes needed to be made. The main idea is not just to finish a project but to mainly build the software that can meet or satisfy the customers' needs or can easily adjust the needs, as the needs and requirements grow over time. Agile can do this by providing updates in short cycles, allowing continuous feedback to be made, to make sure that the product keeps getting closer to what the customer wants (Beck et al., 2001).

The Agile Manifesto, which was written by a group of software developers in 2001, set out four main values and 12 principles. These values and principles mainly focused on people working together, providing a well-rounded working software that allowed them to work with the customers, and to make any changes to their project if needed (Highsmith, 2001). Nowadays, many Agile methods are globally used to help teams develop software more effectively and efficiently. Agile's principles encourage flexibility and instant, honest feedback. This can allow teams to change anything within their project when given crucial feedback from a new customer (Fowler, 2001). Instead of just following a plan, Agile teams can make working software in small parts, which over time can be adjusted if needed. This can make Agile especially useful and impactful in various fields, such as software development, where many technologies and even customers' needs change frequently (Cockburn, 2002). Agile has many benefits that go further than just quick and efficient delivery. As said before, Agile's impact on teamwork with customers and also its team members ensures that the software meets their needs and desires. It also expects teams to be self-organized, where the team members have the space to make their own decisions and work together in creating a standard or impactful project (Larman & Vodde, 2009). This sense of trust that teams give to one another can lead to teams being more inspired, producing more creative and useful ideas, and producing high-quality products.

There are many key characteristics of Agile development. There are many defining characteristics of Agile development that differentiate it from normal approaches. These features help Agile teams to maintain impactful, responsive, and focused delivery of value to the customer that satisfies their needs. Here, we have a few characteristics of Agile development that we will go through:

1. **Iterative Development**: In Agile, development is done in small quantities. Each quantity, normally lasting between one to four weeks, results in a working product increment. At the end of each quantity, the main product is reviewed and tested, making sure that feedback from the stakeholders can be absorbed into the next quantity (Schwaber & Sutherland, 2020). This effective approach allows teams to go

through the customer's needs quickly, rather than waiting until the last minute of the development cycle. The frequent revision at which progress is reviewed also allows teams to make corrections if needed.

2. **Customer Collaboration**: One of the, if not the main focus or idea of Agile is collaboration with the customer. In traditional methods, the customer often has limited collaboration with the development team after the initial requirements phase. However, over the years, the idea of not allowing interaction between customers and staff members quickly disappeared. Agile now highly supports continuous communication with the customer, ensuring that the product can or has met the customers' requirements (Beck et al., 2001). By involving the customer in regular feedback loops, teams can quickly adapt to changes in requirements, priorities, or market conditions. This ensures that the final product is more likely to meet the customer's needs.
3. **Self-Organized Teams**: Agile teams tend to be self-organizing. Rather than waiting for the managers to dictate how the team should function, members of Agile teams are empowered to come to a consensus. This system provides everybody in the team with a sense of responsibility for the project, whether it succeeds or fails (Highsmith, 2001). It is also expected that teams would work together, pool their ideas and talents, and address challenging issues. This will improve decision-making and broader communication competence with respect to the upcoming tasks.
4. **Responding to Change**: Agile also accepts alterations that may occur during the later stages of development. In traditional project management, the alteration of requirements is most of the time viewed as an act of disruption. However, this is not the case in Agile practices (Fowler, 2001). Within Agile teams, they are adaptable and proactive in modifying the project scope and direction as a result of feedback or the changing business landscape. This enables them to develop products that are more relevant and functional to the customers and the stakeholders instead of following the initial requirements that could become unrealistic by the end of the project.
5. **Frequent Deliverables**: In traditional approaches, the product is usually released only at the end of the project. With Agile, the strategy is to update some sections of the project progressively over a period of time. The product team updates the increment at the end of each cycle or sprint. This enables the stakeholders to track the development of the project and provide critique on the completed work (Schwaber & Sutherland, 2020). Moreover, when changes are made, it is easy to notice the revocation of properties, and thus, the chances of experiencing considerable challenges later are low.

SCRUM is definitely the most widely known and used Agile framework. It promotes a great framework for getting work done in sprints by having clearly defined roles, ceremonies, and artifacts (Schwaber & Sutherland, 2020). The SCRUM team includes a Product Owner, who manages the Product Backlog, a Scrum Master, who guides the team in adopting SCRUM practices, and a Development Team that collaborates with the Product Owner to complete the tasks in the Sprint Backlog. SCRUM teams work in sprints, which are short, time-limited periods, usually lasting two to four weeks. At the end of every sprint, the team organizes

Sprint Review and Sprint Retrospective meetings. They assess the progress made, demonstrate the work, and suggest areas that should be improved in the next sprint (Schwaber, 2004).

Kanban is one more Agile framework, though unlike SCRUM in that it doesn't have short-term sprints that are fixed. On the contrary, Kanban emphasizes the visualization of the workflow and controlling the flow of work items (Anderson, 2010). Teams use a Kanban board, which shows the status of work items in different columns like "To Do," "In Progress," and "Done." This visual illustration aids teams in spotting chokepoints and areas that need improvement. Kanban is especially good for projects where processes are closing to complete the job smoothly. Also, the main point is the delivery of work, not the adherence to a fixed set of tasks.

The methodology of Extreme Programming (XP) is destined to the refinement of software and efficiency. XP emphasizes testing perpetually, pair programming, and frequent releases (Beck, 2000). XP intensifies the development process, where the developers and clients work closely together to ensure that the software meets the client's desires. Besides, it highlights the significance of the code being readable and well-structured, which is a must for the project to be smoothly handled over time.

Even if SCRUM is a much-loved framework by many, teams are allowed to use other frameworks according to the requirements of the project. Kanban is mainly used in cases of iterative development and when sprint cards aren't the best option for managing projects (Anderson, 2010).

The introduction of the SCRUM project management methodology, which is a part of Agile, serves as an opportunity for teams to deal with complex work and achieve sophisticated end products in an appropriate time frame. It is a structured method with explicitly defined roles, events, and artifacts that optimize the development process (Schwaber & Sutherland, 2020). The benefits that SCRUM offers include a dynamic mechanism that not only brings the ability to deliver small, incremental changes one by one but also ensures the product is delivered to the customer's needs and adds value quickly.

The pivotal components of SCRUM are:

SCRUM Framework Roles: The Product Owner, Scrum Master, and Development Team are the three main roles in SCRUM. The Product Owner is responsible for the Product Backlog and ensures the team focuses on the most critical tasks. The Scrum Master facilitates the Scrum process and helps the team overcome any hurdles. The Development Team comprises specialists with the knowledge and skills to carry out the tasks in the Sprint Backlog and deliver a working product increment (Schwaber, 2004).

SCRUM Events: The events in SCRUM help ensure that the team stays on track and evolves continually. These events are Sprint Planning, where the team creates the work backlog for the upcoming sprint; Daily Standups, where the team shares their progress and difficulties; Sprint Review, where the team demonstrates the work done in the sprint; and Sprint Retrospective, where the team analyzes the sprint and identifies areas for improvement (Schwaber & Sutherland, 2020).

SCRUM Artifacts: SCRUM applies three main artifacts to plan the team's work: the Product Backlog, Sprint Backlog, and Increment. The Product Backlog is a roadmap for all the features, tasks, and bugs that need to be addressed. The Sprint Backlog is a selection from the Product Backlog that includes tasks planned for completion during the current sprint. The increment is the working product delivered at the end of the sprint (Schwaber, 2004).

SCRUM is a highly effective method for managing complex projects. It ensures that teams remain focused on customer-oriented value delivery while also improving their processes over time.

In conclusion, Agile and SCRUM offer a flexible, adaptive approach to managing software development projects. By breaking projects into smaller, manageable increments and fostering collaboration with customers, Agile allows teams to deliver high-quality software quickly and efficiently (Highsmith, 2001). SCRUM, as one of the most widely adopted Agile frameworks, provides a structured process with clearly defined roles, events, and artifacts that help teams stay on track and continually improve. The benefits of Agile and SCRUM are numerous: increased customer satisfaction, faster delivery times, higher-quality products, and more engaged, self-organizing teams. By embracing Agile principles and implementing SCRUM practices, teams can effectively manage change, deliver value to their customers, and achieve project success in a rapidly evolving development landscape.

Chapter 1: My project proposal

The idea for my project proposal comes from the importance of social networking platforms in connecting people. Blogs and social networking sites are now important tools for sharing ideas and collaborating. To be honest, this idea was comes from Facebook. I would like to create a website similar to Facebook. My plan is to create a simple blogging platform. On this platform, users can post content such as text, videos and pictures. They can also leave comments under posts to discuss and share their views. I want the platform to be easy to use and to help people interact and collaborate. The most important part of my proposal was the blog feature. Users could post anything they wanted, such as personal stories or school projects. I also wanted to make the posts more interesting by allowing users to add pictures and videos. The comments section was another key feature. It allowed users to share their thoughts and discuss posts. This would make the platform a place where users could connect and learn from each other. At first, I only planned to create a simple blog interface because I did not want it to be too complicated. Because of our CS353 team project course is a one-semester course as we do not have too much time for this project, I thought a basic platform would meet most users' needs and be easier to complete within the given time.

I also thought about how to make the platform technically feasible. For the **front-end**, I decided to use **React.js**. Because I have learned that in my previous course. It allows me to apply my knowledge without spending much time learning new tools. In addition, this is a popular JavaScript library that helps build user interfaces. React.js allows developers to create reusable parts, as a result, it is good for making features like posts and comment sections work well together (Facebook, 2013). It also helps create a good experience for users, just like the architecture framework which Facebook uses for its applications.

For the **backend**, I suggested using **Node.js** and **Express.js**. Node.js works well for handling real-time features such as comments and notifications because it is fast and efficient (Tilkov

and Vinoski, 2010). Express.js works with Node.js to help manage server tasks such as storing posts and handling requests. These tools made the backend simple but powerful enough to support the platform. In addition, as I had already studied these tools in my previous coursework, I knew how to use them. This reduced the learning curve and made the project development more manageable.

To store the data, I first suggested using **MongoDB**. This database works well for platforms where the data changes a lot, such as blog posts and comments. MongoDB can store different types of data and is easy to scale when there are more users (Chodorow, 2013). However, some of my teammates did some research on it and after I discussed with them, we decided to use **Firebase** instead. To be specific, firebase allows data to update in real-time, which is good for features like live comments. This was a better choice for making the platform more interactive. Another reason for choosing Firebase was that MongoDB offers limited free storage. We could not complete the project without spending money if we used MongoDB. Firebase solved this problem by providing more free storage and additional features that saved us time.

At first, my proposal was about basic features like blogging and commenting. However, after talking with my teammates, the project has changed a lot. They added many ideas to make the platform better. Together we named it ***CSConnect***, putting together my thoughts and the thoughts of my teammates. It is a social platform for computer science (CS) students at Maynooth University. As we gave it a clear purpose, it becames more useful because it focused on their needs. To be specific, one teammate suggested adding a group chat function. This feature is good for students working on assignments or projects together. For example, they can talk and share ideas in the group. We planned to use **WebSocket** technology to make these chats work smoothly (Fette and Melnikov, 2011). Group chats were hard to add, but they made the platform much better by allowing real-time communication. Also, another teammate suggested a feature for lecturers. This feature lets lecturers post important updates, like assignment deadlines or class changes. It helps students stay informed without checking email or other platforms. In class, we sometimes missed updates because they were announced late. Adding a notification system for such announcements would make communication between lecturers and students better and reduce confusion.

When working on the project, we tried to make the platform easy to use for every student. In conclusion, we chose **Tailwind CSS** to do this. This is a styling tool that makes it easy to create good looking designs (Tuckey, 2017). It also helps the platform work well on both computers and mobile phones. This was important because users may want to access the platform **on different devices**. Mobile responsiveness was an important feature, as many students prefer to use their phones for quick updates and interactions.

Notifications were another important feature in my team member's proposal. For example, users would receive a notification when someone commented on their post. They would also be notified when a teacher made an announcement. These notifications help users stay up to date without having to constantly check the platform. My teammate introduced this idea during one of our meetings and it made a lot of sense. It solved problems we had in the past when class updates were not communicated effectively. Besides, **security and privacy** were also very important. To protect user accounts, I suggested using **JSON Web Tokens (JWT)**. This system ensures that only the right person can log in to their account (Jones et al., 2015). It was also suggested that sensitive data, such as passwords, should be encrypted to make the

platform secure. As users would be sharing personal information, protecting their privacy was essential to building trust in the platform.

Working with my teammates made the proposal much better. My original idea gave the project a good start, but their ideas added more useful features. For example, the combination of blogging, group chats and lecturer announcements made the platform a better tool for CS students. These additions helped us to meet the specific needs of our target users. Planning the project was not always easy. Although we used an agile development methodology (SCRUM), it was hard to balance what we wanted with what we could build in the limited time we had. For example, real-time notifications and group chats were not easy to build. We had to spend time learning about tools like Firebase and WebSockets to make them work. We also had to decide which features were most important. It was tempting to add more features, but we knew we had to focus on the most important ones. This helped us stay on track and finish the project on time.

The reason my project proposal was chosen is that everyone in the team putting their ideas in my plan. My original plan was simple and easy to understand. This made it a good starting point for the team to work on. For instance, one teammate suggested adding a group chat feature to help students communicate. Another teammate advised creating a announcement page where lecturers could share important updates. These ideas fit well with my original plan, and we decided to include them. The proposal became something that everyone could work on together, which made it a shared project, not just my own idea. This step-by-step approach made the project to be manageable.

All in all, the experience of creating CSConnect with my team members taught me a lot. I learned how to plan a project and how to work with a team to improve ideas. Each team member brought unique suggestions and we combined them to create a platform that is useful, simple and specific to the needs of CS students. The development process also improved my technical skills and taught me how to make practical decisions to balance ideas and reality.

Chapter 2:The actual project - CSConnect

Introduction

The team decided to create a social networking website called '*CSConnect*', designed specifically for Computer Science students to connect, share knowledge, and collaborate. The main goal of *CSConnect* was to provide a platform that fosters communication, networking, and community-building among students who share similar academic and professional interests. Recognizing the importance of user experience, the team aimed to build a platform that is both functional and intuitive, ensuring it serves the needs of users in an engaging and seamless manner. To bring this vision to life, the team carefully selected a range of modern technologies, ensuring that each component of the website was optimized for performance, scalability, and usability.

CSConnect also serves as a hub for collaboration, providing features that allow users to **connect with peers, share resources, and discuss academic topics**. Features like posts,

announcements, and private messaging enable students to stay updated on important module information and academic events. The team integrated various communication tools within the platform, ensuring users could easily collaborate on projects and share ideas. By centralizing communication and collaboration in one place, *CSConnect* fosters an active and engaging community of Computer Science students.

Frontend

Frontend of the project was done with **React.js**, which is an open-source JavaScript library maintained by Facebook that focuses on building user interfaces based on components. Development of the solution was made more efficient due to React's modular structure among other things, as developers could easily break the User interface into different reusable components, hence making the development process easier and the product still easy to keep. It is what allows an element of a page to change instantly instead of the whole page having to be reloaded, thus providing the application its best performance in speed of interaction, etc. and the user is provided a seamless interaction.

For the design, the team chose to use **CSS**, a fundamental styling language that allows for flexible and precise control over web page layouts and elements. By writing custom CSS styles, developers can define the exact appearance of elements across the site, ensuring a tailored look. This approach provides complete freedom in design choices, while also promoting consistency throughout the site when combined with reusable CSS classes. With the use of media queries, developers can also implement responsive design to create mobile-friendly interfaces. And to increase accessibility across devices, they use React's Context API to manage state in their components. This lightweight solution efficiently shares state without adding complexity to Redux and other libraries. Making the codebase cleaner and easier to keep,

Backend

For the backend, the team chose **Firestore**, a flexible, scalable NoSQL database from Firebase. Firestore offers real-time synchronization and powerful querying features, which makes it an ideal choice for handling dynamic user data such as posts, messages, and profiles. Firestore's cloud-based structure eliminates the need for a traditional server, simplifying the backend management while ensuring fast and reliable data handling. Firestore's real-time updates also provide a smooth, responsive experience for users, as changes to the data are instantly reflected across all devices.

Database Design

The database structure is designed to support the core features of CSConnect efficiently. Firestore organizes data into collections and documents, ensuring fast retrieval and real-time synchronization across the platform. The main collections include a 'Users' collection to store user information such as profile details. A 'Modules' collection holds data related to academic modules, including module IDs and names. Each module also contains sub-collections for posts, announcements, and messages, allowing students to interact, share updates, and communicate within the context of their specific courses. This modular approach ensures that

the database is scalable and can handle the dynamic nature of the data while maintaining high performance and real-time updates.

The website's user authentication is powered by Firebase Authentication, which provides a secure and easy-to-implement solution for managing user sign-ups, logins, and account management. Firebase Authentication supports multiple authentication methods, including email/password authentication, social media logins (such as Google and Facebook), and anonymous sign-ins. The team integrated Firebase Authentication to ensure a seamless and secure login experience, allowing users to quickly register and access their accounts while maintaining data privacy. It also handles session management, enabling users to stay logged in across sessions, while providing the ability to log out and protect sensitive information. By leveraging Firebase's built-in security features, the team ensured that user data is securely stored and protected from unauthorized access.

Security

The website ensures user privacy by securely handling sensitive information. User data, such as email addresses, is stored in Firestore, while Firebase Authentication is responsible for managing the authentication process, including securely handling passwords. This approach minimizes the exposure of sensitive data and leverages Firebase's robust security measures to ensure that user credentials are stored and processed safely, maintaining a high level of privacy for all users.

Node.js plays a crucial role in the backend architecture of CSConnect, serving as the runtime environment for executing JavaScript on the server side. With its non-blocking, event-driven architecture, Node.js is ideal for handling multiple concurrent requests efficiently, making it particularly suited for real-time applications like CSConnect. This allows the website to provide dynamic, responsive features, such as instant messaging and live updates, without sacrificing performance. Node.js also enables the seamless integration of Firestore, as developers can use JavaScript across both the client and server, streamlining development and reducing the complexity of managing different languages and frameworks. The scalability of Node.js ensures that CSConnect can handle growing user activity and traffic, allowing the platform to expand without significant backend changes.

Version Control

Version control is managed with Git and GitLab using a feature extraction strategy. The team followed a structured workflow where each new feature or bug fix was developed in separate branches, minimizing conflicts and facilitating smooth integration. This approach also allowed for clear tracking of progress and easier rollbacks if needed. The team uses the merge request (MR) for peer review, fostering collaboration, maintaining high code quality, and ensuring continuous improvement. Regular reviews and discussions promoted knowledge sharing and ensured that all team members were aligned on the project's goals and progress.

User Experience

User Experience design played a pivotal role in the development of CSConnect, as the team aimed to create an intuitive and engaging platform for Computer Science students. The

design process began with a focus on understanding the needs and preferences of the target users, ensuring that the website's interface was both functional and accessible. The team employed a user-centric design approach, prioritizing ease of navigation, clarity, and responsiveness across all devices. Key features like streamlined registration and login processes, clear visual hierarchies, and consistent UI elements were implemented to enhance usability. Additionally, attention was paid to the website's visual appeal, ensuring a clean, modern aesthetic that appealed to students while maintaining a professional look. The result was a user-friendly interface that supported seamless interaction and contributed to a positive overall experience for users.

Testing

The team tested the website through interactions and Firestore data insertions to ensure the application functioned as expected in a live environment. User interactions were closely monitored to evaluate the usability of key features such as posting, messaging, and navigation, ensuring the interface was intuitive and the user experience seamless. Additionally, Firestore data insertions were tested to verify the accuracy and efficiency of data handling, including the real-time updates of user-generated content such as posts and messages. This testing process ensured that both the frontend and backend were working smoothly, with data properly stored, retrieved, and synced across users in real time. This hands-on approach was helpful in identifying usability issues and improving the user interface for a smoother experience.

Development Challenges

Throughout the development of CSConnect, the team encountered several challenges when working with new technologies and frameworks. One of the primary difficulties was learning to integrate Firebase Authentication and Firestore into the backend, as the team was initially unfamiliar with Firebase's structure and real-time database capabilities. Understanding how to efficiently manage data synchronization and ensure consistent performance across devices required a steep learning curve. Additionally, implementing real-time features, such as live chat, posed technical hurdles due to the complexity of handling asynchronous events and data updates across multiple users. On the frontend, working with React.js introduced challenges related to managing component state and effectively leveraging React's lifecycle methods. However, through collaboration, extensive research, and trial-and-error, the team overcame these obstacles, gaining valuable experience and ultimately building a robust, scalable platform.

Chapter Summary

In summary, building CSConnect was a challenging yet highly rewarding project that required thoughtful planning and careful selection of technologies. The team chose React.js for its component-based architecture, which allowed for efficient and scalable development of the frontend. For styling, custom CSS was used to ensure a tailored, consistent, and responsive design across devices. On the backend, Firebase and Firestore were chosen for their scalability, real-time data synchronization, and seamless integration, enabling efficient handling of user data, posts, messages, and more. The platform was designed to be accessible on a variety of devices, ensuring a smooth user experience across both desktop and mobile

platforms. The development process was driven by extensive testing, including interaction testing and Firestore data handling, to ensure optimal performance and a bug-free user experience. Effective version control practices with Git and GitLab, including feature branching and peer review, ensured code quality and facilitated smooth collaboration. The team's strong communication and collaboration, supported by robust tools and agile methodologies, played a crucial role in overcoming challenges and ultimately delivering a reliable, user-friendly platform that meets the needs of Computer Science students.

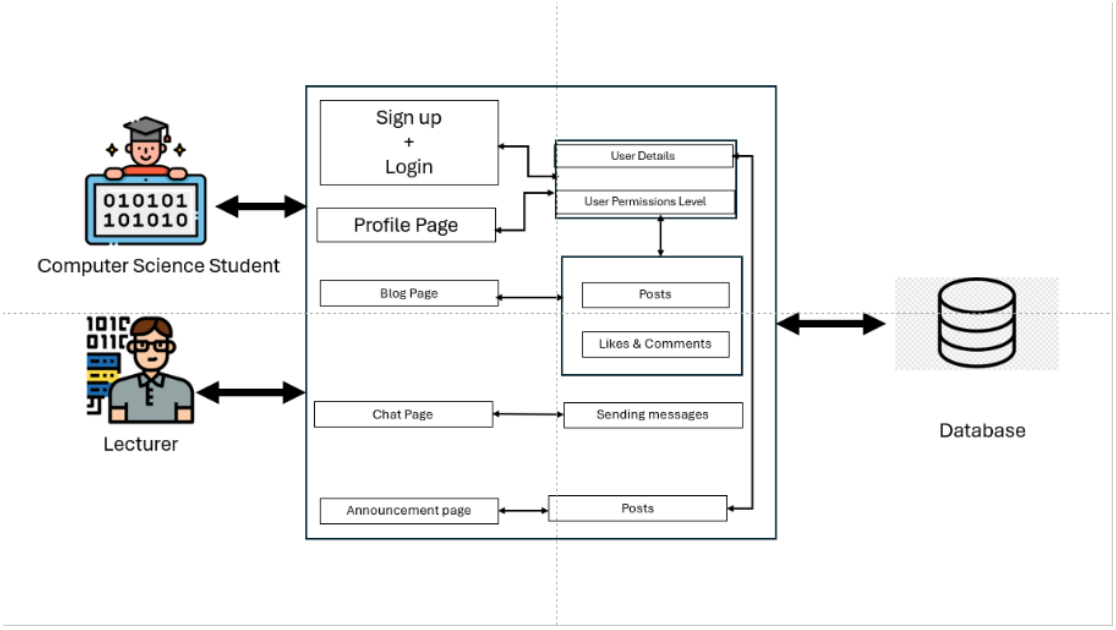
Chapter 3: The SCRUM process

Initial Scrum meeting

The initial scrum meeting was a very key moment in us laying down the foundation for CsConnect's development. This meeting began with the team trying to develop a sprint goal however we were unable to officially pick our project in the first week due to team members being unavailable with valid reasons such as illness and interviews this was a challenge but we were able to overcome this with effective communication through our team channel. So, with the team members that were present in week 1 we decided to go with Emmanuel's proposal which was to create a central platform that students could communicate and collaborate on. It was a great idea as it implemented the other proposals presented and would allow everyone to get involved as much as possible since everyone in the team possessed skills that would make it possible to create such a product. Since this was a mixture of more than one team members project proposal the team posted the proposal into the Teams channel and communicated that we would create the features using a collection of user stories, it was prompted by the SCRUM master at the time that everyone should create the user stories based on features that they would want implemented into the application and we would discuss it in the next Scrum meeting.

Graph 1: Project Proposal

Due to us being behind a week our first we held a meeting outside class time which allowed the team to make sure we were on course with completing the user stories this meant we would be able to achieve building the product in the set time given to us. We each individually presented our user stories describing further why they chose the feature and how this could aligned with their vision of CsConnect, allowing everyone to explain their vision gave the team the opportunity to collectively narrow down who the target audience was and how such a platform would be able to help the target audience. At the end of this meeting, we were able to create a story map which you can find more details on in the story map section.



Graph 2: Target Audience & Structure Diagram

Project Estimation

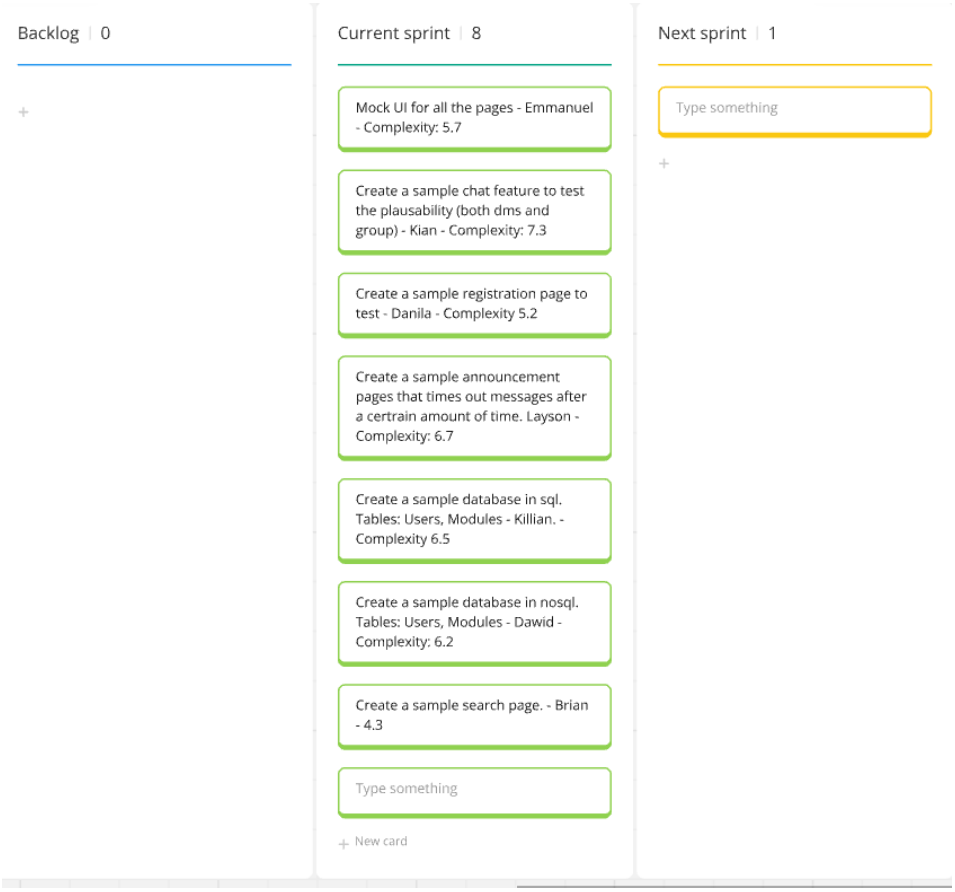
Accurate job estimates were essential for effective emergency
For estimating the tasks, we used the Wideband Delphi method. This involves :

1. Assemble a group of experts (around 10)
2. Give each estimator a deck of cards (usually 0, 1, 2, 3, 5, 8, 13, 20, 40, and 100)
3. Moderator reads a description of the user story. The product owner can answer questions from estimators.
4. Each estimator selects a card and places it faces down on the table. When all estimates are in, the cards are flipped over.
5. If the estimates vary widely, the owners of the high and low estimates discuss the reasons why their estimates are so different. All estimators should participate in the discussion.
6. Repeat from step 4 until estimates converge to within some predetermined threshold.

We used planning poker to estimate the size of each task. We used a measurement unit of 0-10 to measure each task, 0 meaning that it wasn't a difficult task to complete and 10 meaning that it would be very difficult to complete. Some of the things we made sure to consider when giving our rating included:

1. **Team level skills:** How complex would it be to complete the task – Is there a learning curve that must be overcome?
2. **Time:** How long will it take to implement the task? – Some of the tasks may consume much more time to complete than others.
3. **Dependencies:** Does this task rely on other tasks being completed first or does it have any other dependencies that could bring complications.
4. **Iteration:** Will the task require multiple iterations or feedback from other team members before it could be completed?

Having these sorts of factors as well as many others allowed the team as well as each team member to accurately estimate the complexity of each task. The categories we voted on a Sign up and Login, Announcement, Posts, Comments, Administration and Messages. This method was useful as it allowed the team to split the tasks evenly and gauge how capable and comfortable everyone was with completing their tasks. We made sure that as a team that there was a collective agreement on the rating of each of the tasks and if there were any anomalies we allowed the team member to explain their reasoning on why they gave their rating. If a certain team member thought they wouldn't be able to complete the task at all due to reasons such as the skill complexity of the task or due to the reason that it was too time consuming we were able to swap the tasks between team members this happened throughout each sprint as we understood that some people were still trying to secure internships while some others had already secured one this was a great display of teamwork and through this we were able to stay on track of completing our tasks during the sprints, this was an important step in the sprint planning as this was not only about dividing the tasks amongst each other but also ensuring that everyone understood the sprint goals and their role in achieving them. The output of the sprint planning was a product backlog, which detailed a list of tasks each person was to complete during the sprint along with an estimate of its complexity.



Graph 3: Product Backlog

Graph 4: Product Backlog Progression

As you can see from the two images there was a lot of progression in between sprints this would all be done during our Sprint Review and Sprint Retrospective which included:

- The team presenting what it accomplished during the sprint this helped us understand the progress we made towards the projects goal.
- Unlike our Scrum meeting instead of just talking about our completed tasks the team typically provided a hand-on-demonstration of new features, functionalities or architectural improvements.
- There were no slides, it was more of an informal nature as the focus was on collaboration, open discussion and feedback rather than conforming to strict presentation formants.

All of this meant that as a team we could reflect on processes, tools, communication and teamwork to identify strengths and areas for improvement. To plan our next sprint we would use the Stop/Start/Continue method in using this method it meant we would discuss and identify any new tasks or practices that would improve our sprint process and would maintain our goal of completing the project on time but we would also discuss if there was anything hindering our progress deciding to eliminate them or continue them by possibly breaking down the task or sharing the task amongst other team members.

Meeting minutes

Scrum practices involve taking down our talking points as well as any important information that takes place during the minutes this allowed us to be clear with the minutes, track our discussion topics and follow through with decisions made along with specific assignments of tasks.

In the early phases the minutes were much longer which could be pointed back to us not being able to pick our proposal in our first week. We were able to overcome this problem

with lengthy discussions and collaboration. These eventually became shorter as time progressed and we were able to maintain a consistent time.

Backlog :

Team updated the backlog based on new tasks and completed ones. Team members took tasks that were similar to their sample tasks from last week. Team re-estimated the complexity of each task based on team feedback.

Gitlab :

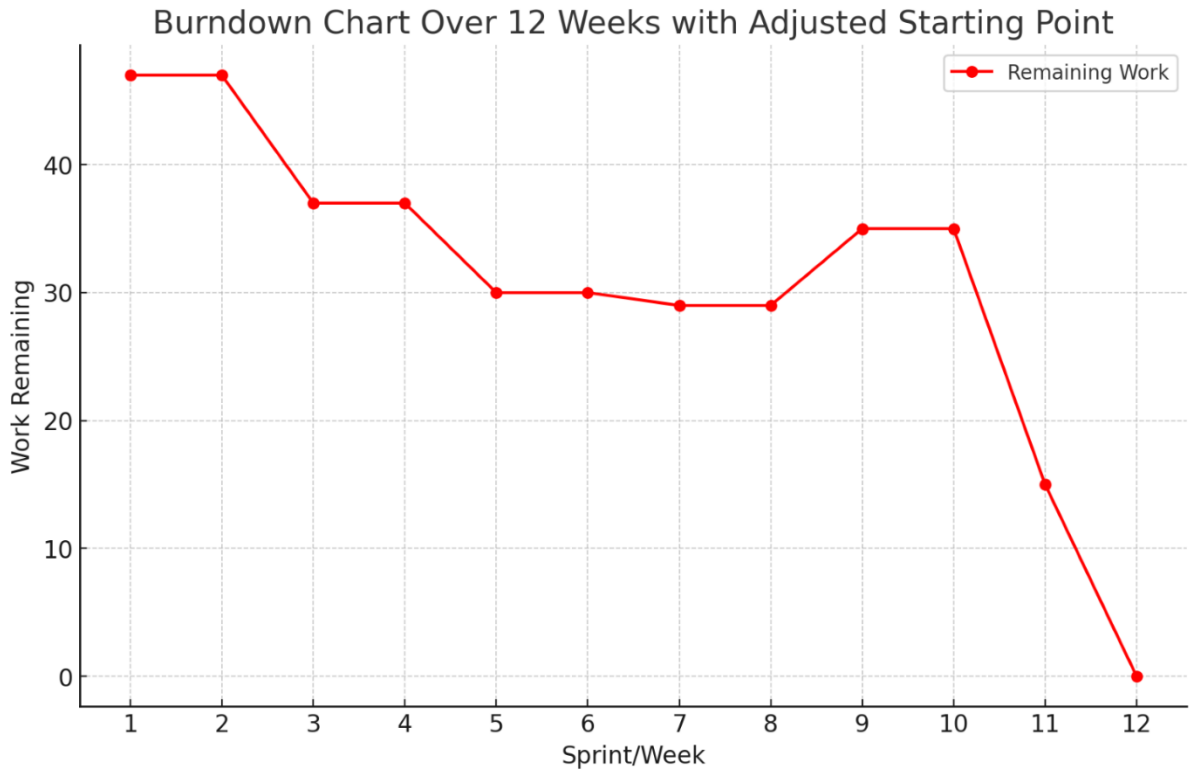
Danila setup the gitlab for the team and invited everyone.

Kian setup the basic project on gitlab with node js, react and firebase.

Each team member sets up their own branch for their code, team will merge code later when agreed.

Graph 5: Minutes

Burndown chart



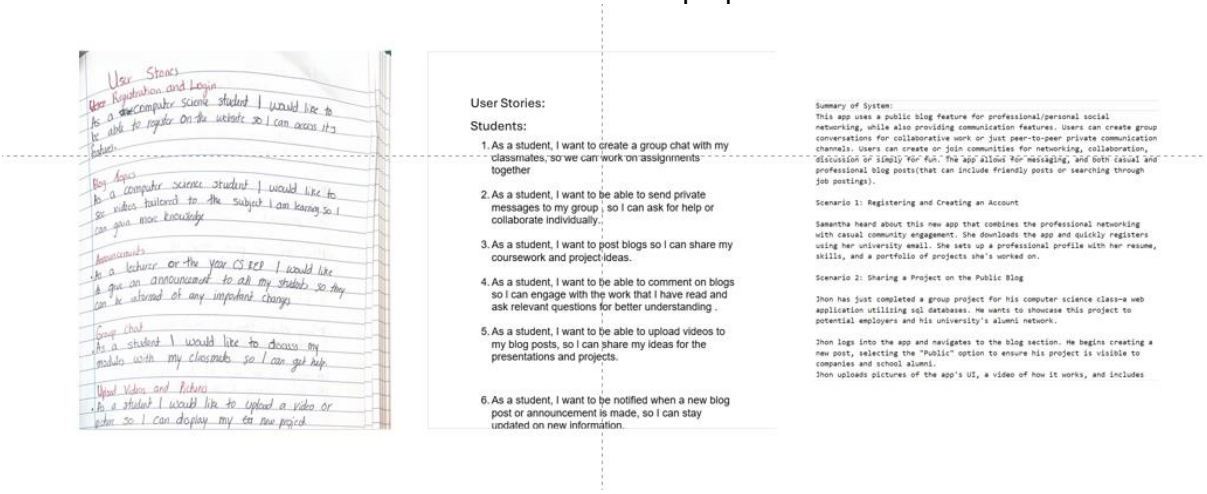
Graph 6: Burndown Chart

We used a burndown chart to estimate our and track our progress over the 10 weeks. This chart shows the total workload left while also helping us measure our progress. We modelled this using the example in the lecture notes.

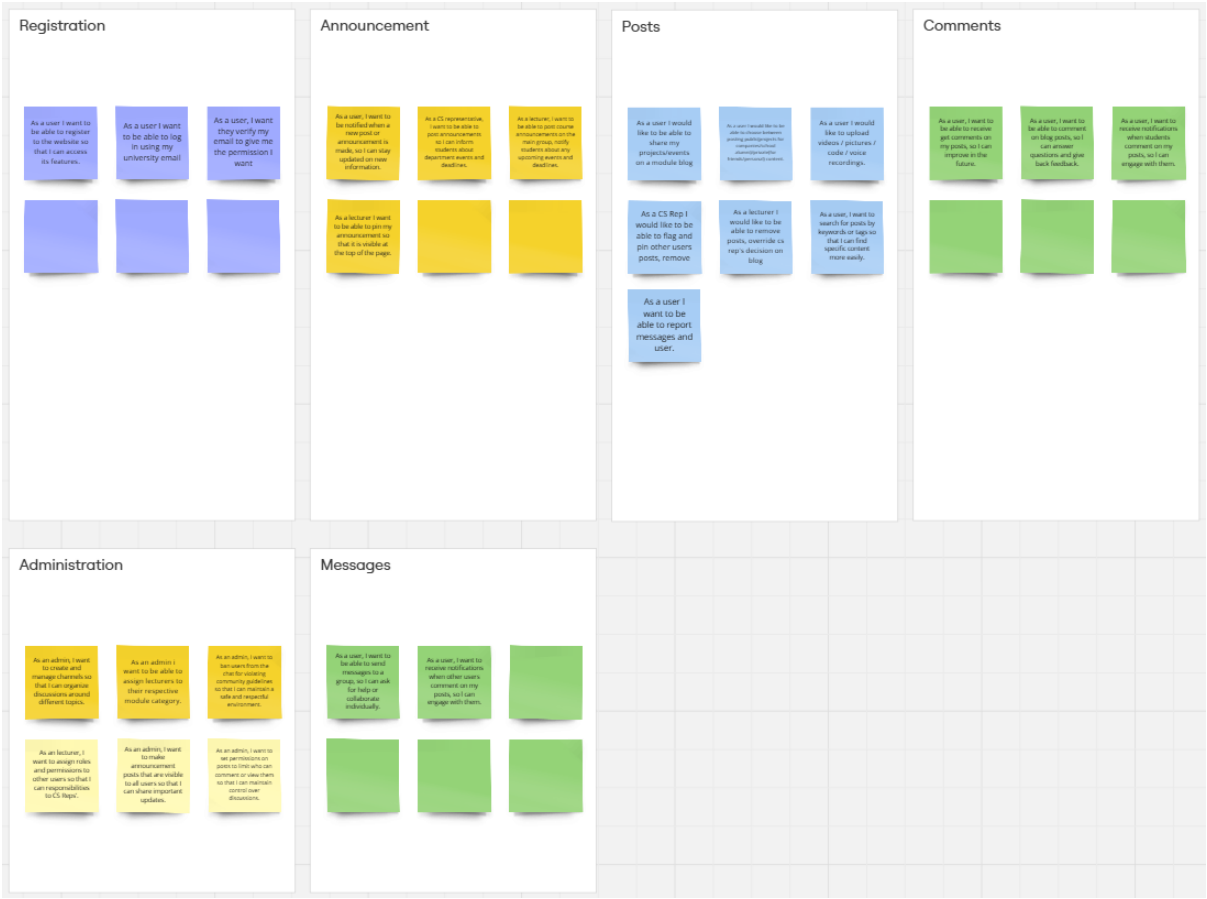
This chart was especially useful because it gave us a constant visual of how on track, we were to complete the project. You can see that in the early weeks the work remaining was quite high this was due to us not choosing our project in time and thus leading to us working more hours. The work that was needed to be complete became less overtime however it picked up again in the last few weeks, this was due to certain features not working when we merged the pages together. This was frustrating and challenging but we were able to meet up constantly outside of class time to try and solve these issues. One of the things we did as a team was also break up into smaller teams to try and overcome these tasks for example the team member that was working on the Sign in and Log In (Danila) would work together with the team member that worked on the database (Dawid). Another example is the person that worked on the UI/UX(Emmanuel) had to work closely with the other team members merging the pages (Kian) and other team members that worked on some of the frontend while also doing backend (Songyan)(Brian)(Killian). Overall, this chart enhanced our efficiency, improved our talk management and helped us stay focused and adaptable while achieving our project goals.

Story maps

The story map was an important step in organising our project and aligning our team's vision. We each individually presented our user stories describing further why each team member chose the feature and how this would align with the teams vision of CsConnect, allowing everyone to explain their vision gave the team the opportunity to narrow down who the target audience would be and how such a platform would be able to help the target audience. This also allowed a collective agreement on certain features, any overlapping ideas and a collective disagreement on certain features, one example of this was making the pages customisable even though this was a great idea, and we could've possibly added this to our story map we were able to agree as a team that this was not a priority due to our time constraint. The sharing of our user stories also gave us an opportunity to implement any features that would make our platform unique from other existing platforms. At the end of this meeting we were able to create a story map. Dawid was able to combine everyone's user stories together and by doing this we were able to agree that the features that were frequently mentioned would be priority to the user audience also top priority to us as tasks that we should focus on first in our backlog. In terms of making our product unique we made live scenarios of user stories since the team be was made up of a group of computer science students we believed we would have the knowledge to know what unique features would help us in terms of social interaction and also educational purposes.



Graph 7: User Stories



Graph 8 Story Map

By using the story map, we were able to define clear sprint goals using our backlog which we also created on Miro. The story map provided a visual guide that would allow us to track our progress through the development process. This collaborative process allowed the team to be aligned and well organized to meet the tight deadline.

Testing in the SCRUM Process

Testing is a key part of software development, especially in SCRUM. It ensures features work correctly and helps catch issues early. In SCRUM, testing often happens during sprints, where team members test their code locally before integrating it into the main repository. Common methods include unit testing for individual components and integration testing to ensure modules work together. This process allows continuous improvement and aligns with SCRUM's iterative approach (Cohn, 2010).Our SCRUM meetings provided a structured and effective platform for testing the code we were developing and synchronising team progress. Testing played a vital role in ensuring that all features worked and in identifying potential problems early on. Each team member started by **testing their code locally**. This preliminary step helped to catch simple errors such as syntax errors or minor bugs in functionality. If the code performed as expected, it was committed and pushed to the Git repository. When problems arose, team members **first attempted to resolve them independently**, sometimes with AI tools like Copilot or ChatGPT. For instance, when a team member encountered mismatched brackets or incomplete logic in a function, AI tools provided suggestions that saved time and allowed us to focus on the project's broader objectives.

A significant obstacle arose during the integration of the front-end and back-end components into a cohesive system. While testing API calls, a front-end developer discovered persistent errors indicating incorrect parameters. Despite multiple attempts, the issue remained unresolved at the individual level. To address this, both front-end and back-end developers **collaborated during SCRUM meeting**. Together, they analyzed the code from both perspectives and identified a mismatch in how data was transmitted and received. By refining the mechanism for external API calls, the team successfully resolved the problem. Another notable challenge occurred during the development of the login functionality. At one point, the API responsible for authenticating users became entirely unresponsive. Despite numerous discussions and attempts to debug the issue, the team could not identify a solution. Ultimately, we sought assistance during the Wednesday CS353 lab. With the support of the teacher, the root cause was identified, and an appropriate fix was implemented.

All in all, testing was not solely about detecting and resolving errors; it also fostered collaboration and enhanced the team's understanding of the project. We working together to overcome technical challenges, and developed a stronger sense of camaraderie and a deeper insight into the project's architecture. This also helped us learn a lot.

Future Releases/Sprints

At the moment our platform is only for the students of Maynooth University. So the future work would be to **make our platform ready for a wider market and usable for students in all universities.** Therefore, we need to do more work after the current version. We should improve the system to support more users. We should also **add a better user registration system** so that students from any university can easily join the platform. For example, we can allow universities to choose their university when they register. Also, professors from different universities can customise the lecturer announcement feature. They can use the platform to share information specific to their university courses.

We also need to improve the **group chat feature** to better support larger user groups. At the moment, the group chat is quite basic and works well for small teams collaborating on projects. However, as more students start to use the platform, we need to ensure that it can handle larger groups effectively. One improvement could be the addition of a search function. For example, users could quickly search for news related to their university, specific courses or ongoing projects. Imagine a student working on a large group project needing to find a discussion thread from weeks ago - this feature would save time and reduce frustration.

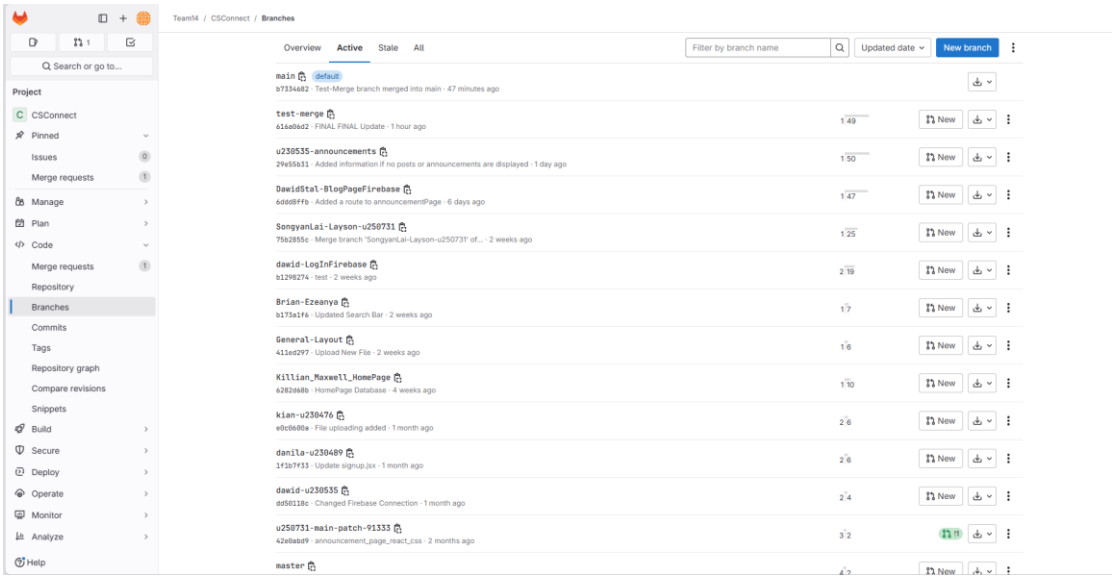
Another important improvement is the **upgrade of the notification system**. Currently, users receive notifications when someone comments on their post or when a lecturer makes an announcement. While this setup works for now, we can make it more personalised and user-friendly. For example, users could set their notification preferences to only receive notifications about topics, group chats, or updates they are most interested in. This would not only prevent users from being bombarded with irrelevant notifications, but also help them focus on important updates, making for a more productive and less distracting experience.

Ultimately, the most important step is to prepare the platform for **real-world use**. This requires rigorous testing to ensure that the system works smoothly under high user demand. For example, as the platform scales to accommodate thousands of students across multiple universities, it must maintain stability and speed without crashing or lagging. User security is

also a top priority. We should **enhance the login system** with features such as two-factor authentication and implement advanced encryption methods to protect sensitive data such as user profiles and chat histories. To refine the platform, we could starting with **small-scale testing**. For instance, we could introduce the platform to the computer science department at Maynooth University. This initial roll-out would allow us to gather valuable feedback from students and lecturers who are actively using the system. Based on their input, we can identify and address any bugs or usability issues. Once the platform works seamlessly within this department, we can extend its reach across the University, following a successful implementation at Maynooth University, then we could strategically roll it out to other universities in Dublin, building momentum as we go. This gradual expansion would allow us to monitor performance, ensure reliability and adapt to the needs of a growing user base. Once the platform is proven to be stable across Dublin, we could roll it out to other cities, then nationally, and eventually to universities worldwide. By taking this step-by-step approach, we can create a reliable, user-focused platform that truly meets the needs of students and educators at every stage of growth.

Code Versioning

For this project we used **GitLab** to do code versioning. It is a web-based DevOps lifecycle tool that provides a Git repository manager, CI/CD pipeline features, and more. GitLab allows teams to collaborate on code, manage projects, and automate the software development process. GitLab supports the entire DevOps lifecycle, from planning and source code management to CI/CD, monitoring, and security (Smith, 2023). The GitLab we used is typically used by the Computer Science Department at Maynooth University for code version control. We started by creating a shared repository to store our source code. We used branches to manage different modules and versions. However, only two members of the team knew how to use GitLab at the start. So they spent some time helping the rest of us learn. Soon everyone understood how to use Git and downloaded the Github desktop. Then they created their own branches(see Graph 1). This was very important because it allowed our team to become familiar with version control early in the development process. For example, when team members made changes, they submit them to their personal branches and everyone can see it. After that, we decided whether to merge the changes or reject some of them. Sometimes, we had conflicts while merging the code. In these cases, we worked together to resolve the conflicts. We also faced situations where a team member accidentally broke a branch, but we could fix it quickly because we had backups. These issues did not happen often since we discussed tasks in our meetings to avoid overlap.



Graph 9: Team GitLab Branch Screenshot

There were also some common mistakes during development. For example, some team members forgot to include the *.gitignore* file. As a result, they uploaded large folders like *node_modules* to the repository. This was a problem because these folders can be generated offline and take up too much storage space, which slows down downloads.

Towards the end of the semester, we created a “test-merge” branch. This branch was used to combine everyone’s code and test it as a complete system. Overall, Git and GitLab played an important role in our project. They allowed us to manage and track our code efficiently. Although we made some mistakes, this project gave everyone valuable experience using Git for team development.

In conclusion, our team used GitLab to manage the code effectively. We successfully completed several iterations of the code. By using branches and merging strategies, we collaborated well to develop different parts of the project. Our project link is *https://gitlab.cs.nuim.ie/team14/csconnect*, which shows the results of our teamwork and code management.

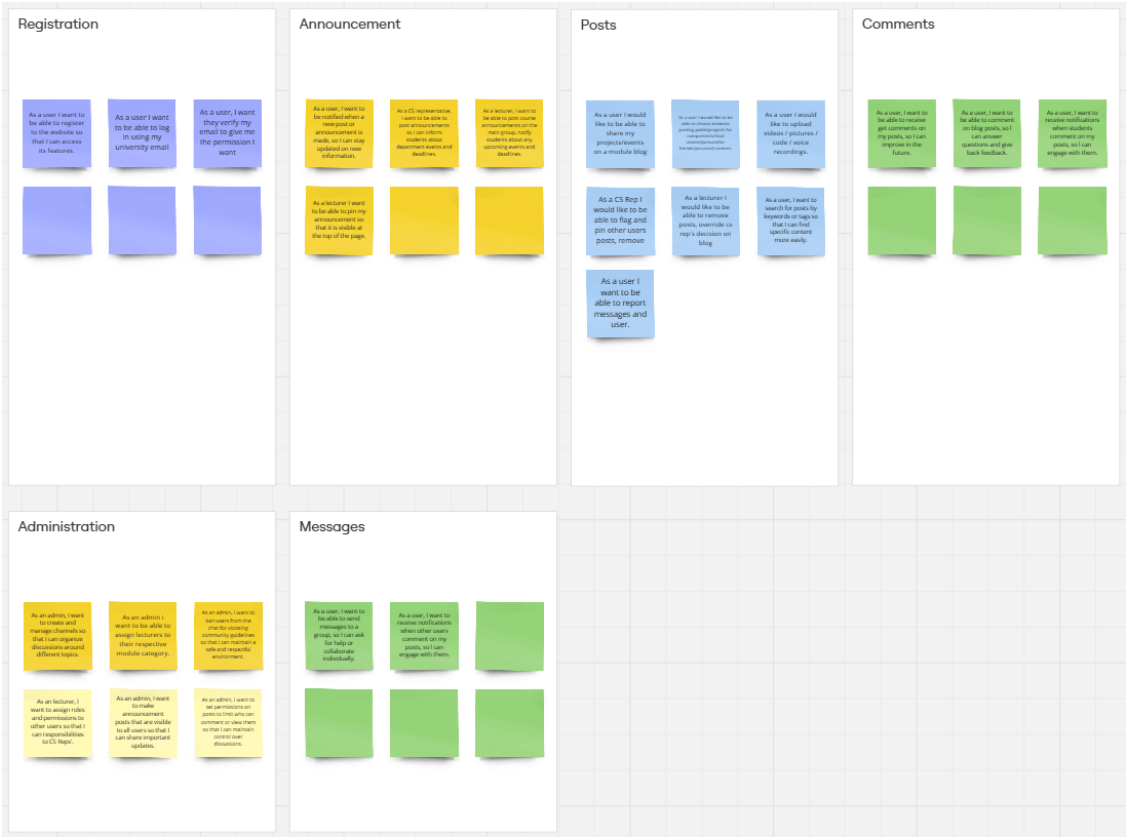
Team Communication/Management

At the start of the project, none of us knew each other. Our first meeting did not go well because some team members were busy with interviews or hospital visits. However, this was not a big problem. The students who attended the meeting shared all the information with the others later. Even though we missed some members, it was clear from the beginning that everyone wanted to create a high-quality project.

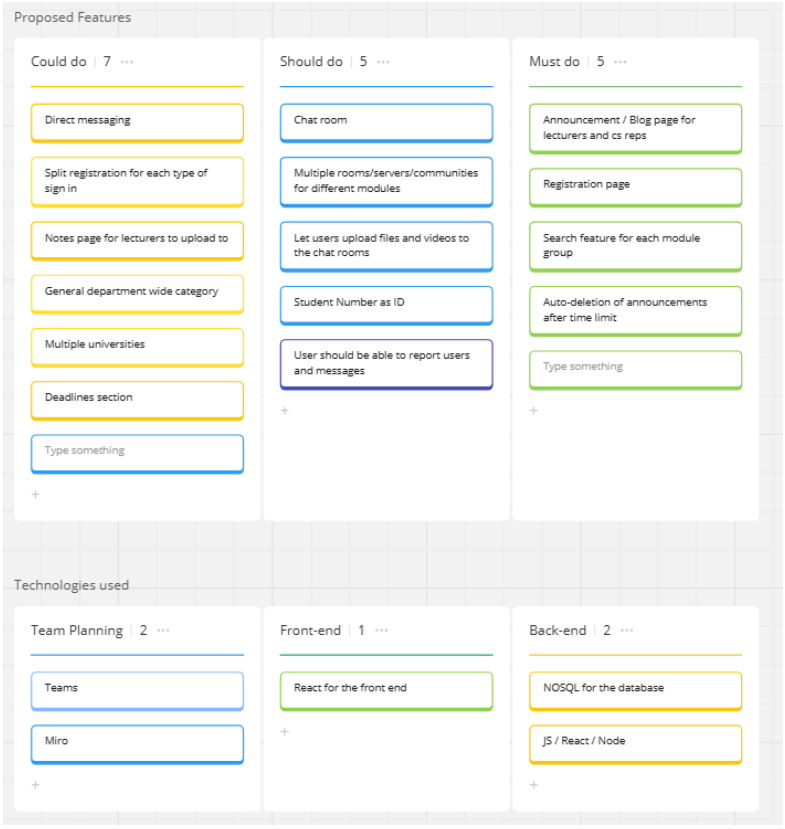
In our team, no one had experience with agile development or SCRUM. We learned about it in class and applied it to our project quickly. **Every two weeks, we assigned a new Scrum Master and a note-taker.** The Scrum Master organized meetings by asking three simple questions: **What are you doing? What will you do next? What is blocking your progress?** These questions form the foundation of the Daily Scrum, which focuses on maintaining

communication, identifying challenges, and keeping the team aligned toward shared goals (Schwaber & Sutherland, 2020). This structure helped the team stay organized and made sure everyone knew their tasks. By rotating the Scrum Master role, every team member had the chance to facilitate and develop leadership skills. This approach aligns with the agile principle of fostering collaboration and shared responsibility (Beck et al., 2001). Additionally, assigning a **note-taker** ensured that meeting discussions were well-documented, which helped team members who could not attend stay updated on the project's progress. The role of the note-taker was shared equally. Every two weeks, a new team member was responsible for recording meeting notes. This made the process fair for everyone. In our team, communication was never a problem. We talked openly in meetings and shared updates on Microsoft Teams. Team meetings were always positive because everyone knew we were working together to create the best project possible.

We also used **Miro** to organize tasks. Miro is an online collaborative whiteboard platform designed to support real-time teamwork, brainstorming, and project management (Miro, 2023). It provides various templates, tools, and frameworks that help teams visually organize ideas and tasks. Miro was very useful because it allowed real-time collaboration. For example, we could all work on the same board at the same time, adding and updating tasks instantly. We started by listing all the features we needed to implement (see Graph 2). After that, we categorized them by priority level: "**must do**," "**should do**," and "**could do**" (see Graph 3). Finally, we broke these tasks into weekly sprints, ensuring that each week we had a clear set of deliverables. This step-by-step breakdown helped us stay organized and focused throughout the project. This feature ensured that everyone could contribute and stay updated, no matter where they were working from. Even though it was our first time using Miro, we learned it in a short time and manage our work efficiently.



Graph 10: all the features we needed to implement in Miro

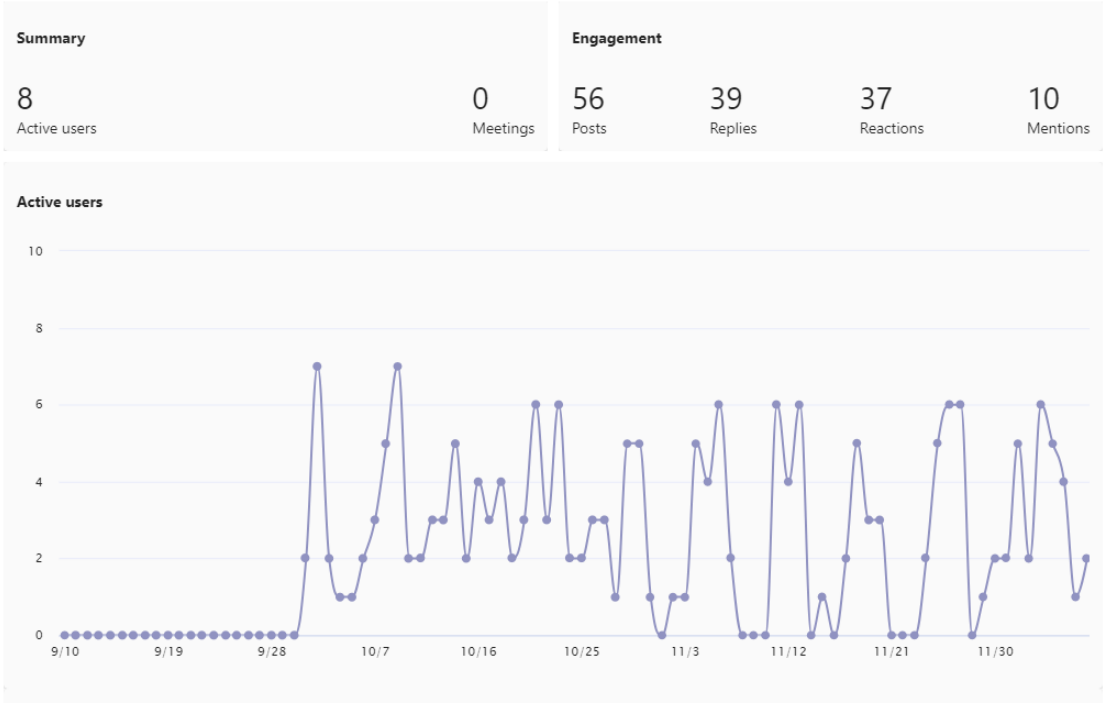


Graph 11: proposed features and techniques used

Remote Working

Microsoft Teams is a communication and collaboration platform developed by Microsoft. It combines features like chat, video calls, file sharing, and document collaboration into one tool, making it popular for both educational and professional teams (Microsoft, 2023). The university offered each of us in the group a Teams account. It made remote work much easier. And most of us had used Microsoft Teams before because lecturers often used it to communicate with students. However, we had never used it for group projects before. Fortunately, this was not a problem. Our team used it for remote communication and GitLab for managing the code repository.

During our SCRUM meetings, we assigned tasks to team members. **Microsoft Teams** allowed us to update each other on the progress of tasks, report any problems, and confirm when tasks were complete. When someone finished a task, they would notify the team to pull the latest version of the code from **GitLab**. This made communication smooth and kept everyone working on the latest version of the project. Since the launch of the CS353 Team Project, all team members have remained highly active on Teams (see Graph 4 and 5).



Graph 12: Active users in Teams

Graph 13: Engagement in Teams

Teams also allowed us to collaborate on documents. For example, we wrote meeting notes together in Word documents, taking turns to record minutes. As mentioned earlier, GitLab was used for managing the code. At the start, we spent time learning how to use Git for remote work. Once everyone was comfortable, it became a very effective tool. Team members could push updates to GitLab, and Teams would notify the group about the changes. This constant communication meant we did not waste time waiting or working on outdated versions of the code.

Remote Working AI Code Generation Experience

AI tools played a small role in our project development. Although they were not used much, they were still helpful for fixing errors and debugging. AI was especially good at finding small syntax or logic mistakes, such as incorrect brackets or function declarations. This saved us time because we did not have to spend hours looking for small problems.

We also used AI to help generate some CSS code. This reduced the amount of repetitive work and allowed us to focus on other tasks. However, AI had its limitations. Sometimes, it could not detect problems in our project because the code was spread across many files. Even when AI gave answers, they were not always correct. In some cases, it provided the same incorrect solution repeatedly.

Because of this, we could not rely completely on AI. For large projects like ours, AI still has many limits. We spent a lot of time testing and debugging the code ourselves. While AI was useful for small tasks, it could not replace human effort.

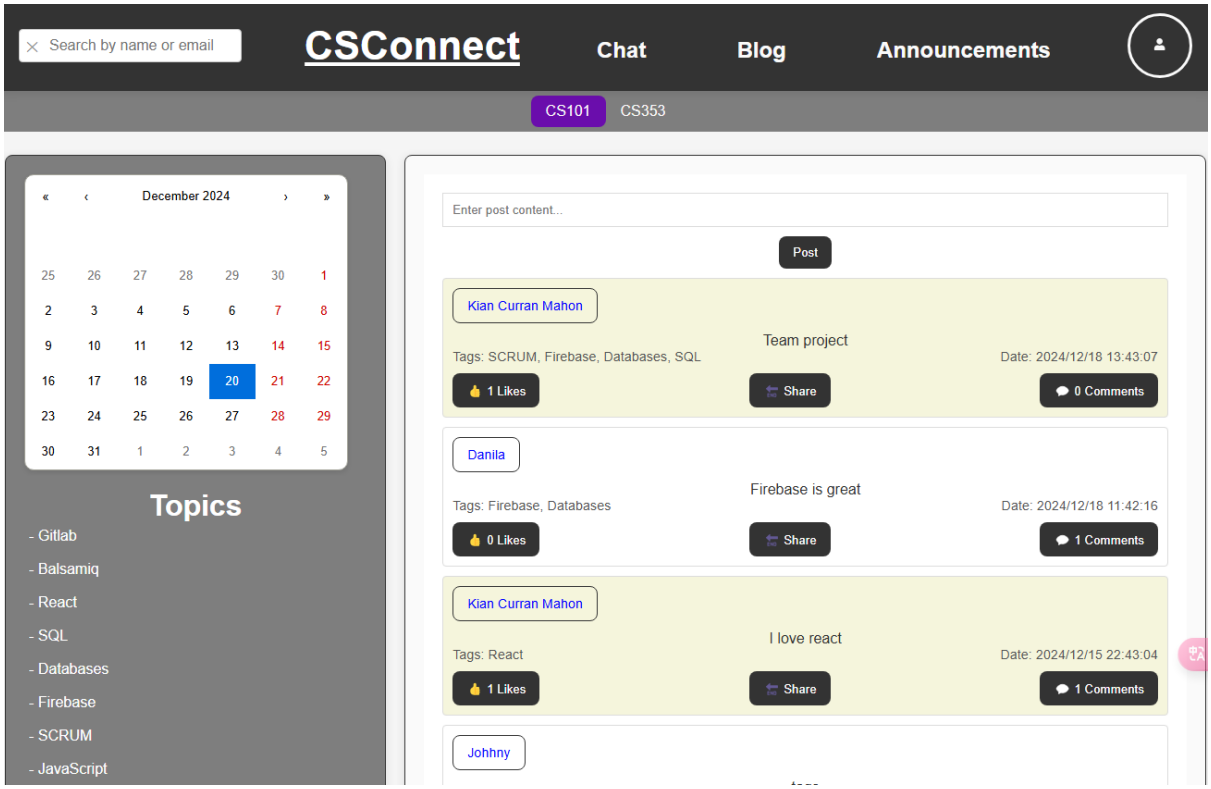
Chapter 4: My contribution

During the development of the CSConnect platform, my contributions included critical areas such as the **blog page**, **announcement page**, and **user profile interface**. Although my major contributions were in front-end features, I also supported Dawid in setting up and

troubleshooting database connections. **My role involved creating user-friendly interfaces, ensuring smooth functionality, and addressing technical challenges as they arose.**

Blog Page

The Blog Page was one of the central features of the CSConnect platform and played a very important role in fostering collaboration among students.This was also the most substantial part of my work on the team project. **My key responsibility in this area was to design, implement, and ensure the functionality of the blog system.** This should have an easy-to-use interface where students can share posts, discuss comments, and interact with other students while seamlessly integrating all the technical complexities.



Graph 14: The blog page

1. **Creation of posts and their display**
    - The blog page allows users to create posts with content and associated tags. For instance, users can tag a post as "React," "Firebase", and "SCRUM" to further make the tagged post show for other users looking for such topic interests.
    - To make it more user-friendly, I created a dynamic input field to add new posts. Upon input of content and clicking the "Post" button, this sends data to Firebase Firestore. This ensures all the posts are saved securely and can be retrieved later.
2. **Dynamic Data Fetching**
    - I wrote the logic to fetch blog data from the Firebase database whenever the page loads or a new post is added. The useEffect hook maintains the blog page

up to date in real time and makes sure the posts are returned in reverse chronological order—from newest to oldest.
    - Each post displays the author's name, the content, associated tags, time and date posted, along with the number of likes and comments.
3. **Select Tag and Filtering**
    - The ability to add tags is a main feature of the blog page. Every time a user makes a post, they have an option to select one of the predefined tags such as "React" or "Node.js." These tags help in filtering posts and make them more accessible for other users.
4. **Comment System**
    - Commenting functionality was implemented to enable users to comment on posts. Comments are stored in the database and fetched dynamically to update the page in real time.
    - I added the functionality of showing the author's name and date of every **comment** to enhance readability.
5. **Like Functionality**
    - Users can like posts to show their appreciation. The number of likes changes dynamically based on user interactions with the posts. For this feature, coordination with the database integration was necessary so that the likes would save without erasing other data.
6. **UI/UX Improvements**
    - I prioritized making the interface intuitive and visually appealing. Features like avatars next to the author's name, a clean layout for posts, and a responsive design ensure that the blog page works well on all devices.

Technical Implementation

The blog page was developed in React; it allows me to utilize component-based architecture for maintaining parts of the page. I divided the blog page into three major components:

- **Sidebar**: Shows a calendar and a list of selectable topics
- **Post Area**: This area includes an input field to add posts and will visualize a list of posts that exist.
- **Individual Post**: This handles the displaying of each post, its respective comments, likes, and other metadata.

To manage data, I integrated Firebase Firestore as the database.This work was completed together with my teammates. One of the technical challenges was how to handle the timestamps that were coming from Firestore. Firebase stores timestamps in a peculiar format, which needed to be converted into a human-readable format before being displayed on the blog page. **I resolved this by using JavaScript's toLocaleString() method, ensuring all timestamps were displayed in a consistent and user-friendly way.**Another challenge involved ensuring proper synchronization between the front-end and the database, especially for features like likes and comments. For example, when a user likes a post, the system checks if they have already liked it to prevent duplicate entries in the database. **This logic was implemented using Firebase’s arrayUnion and arrayRemove methods.**

While I primarily focused on the blog page, its development required collaboration with other team members. Dawid worked on the database configuration, ensuring that data storage was

secure and efficient. **I assisted him by testing database queries and providing feedback on data structure decisions.** Additionally, **I worked closely with the teammate** responsible for user authentication to link blog posts and comments to specific users.

Challenges and Solutions

1. **Real-Time Updates**
    - **Problem**: Ensuring that new posts and comments appeared instantly without requiring a page refresh.
    - **Solution**: I used Firebase listeners to fetch and update data in real time. This allowed the blog page to reflect changes as they happened.
2. **Tag Management**
    - **Problem**: Allowing users to select multiple tags while ensuring the tags were properly stored and displayed.
    - **Solution**: I introduced a toggle functionality whereby the user can add or remove tags by clicking on them. The selected tags were stored as an array in the database.
3. **Commenting System**
    - **Problem**: How to show comments in real time and link them to a user's profile.
    - **Solution**: I wrote logic for fetching user data dynamically while showing comments. This included displaying the commenter's name and avatar alongside their comment.

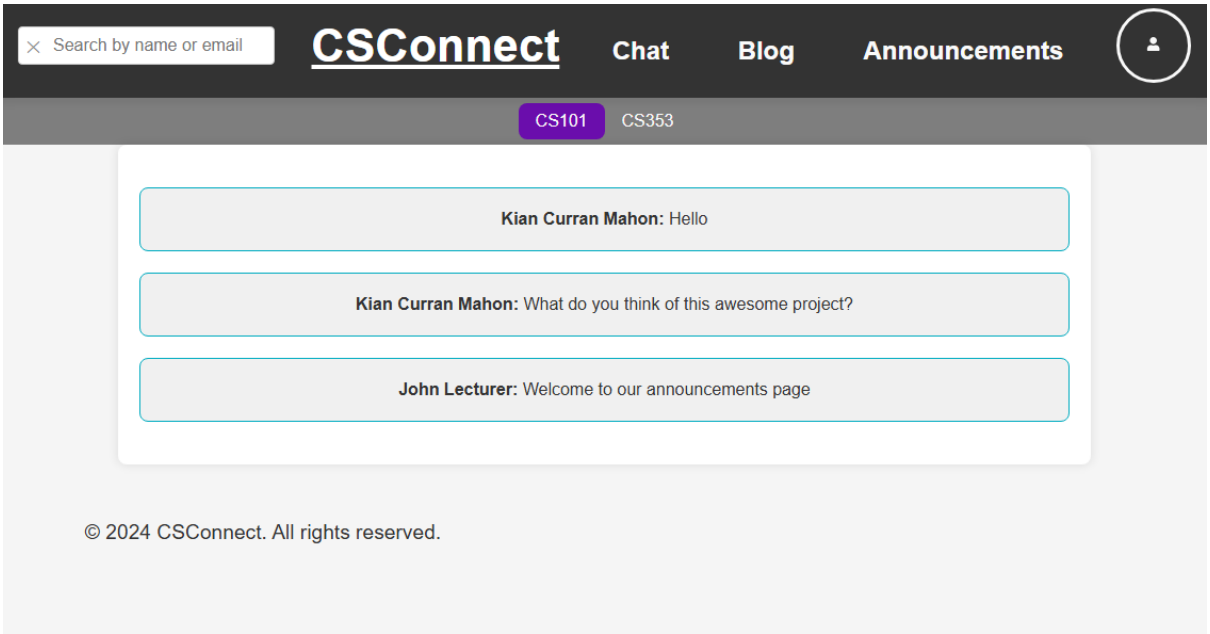
The blog page became one of the most engaging features in CSConnect. It provided that place where students shared their knowledge, discussed ideas, or collaborated on projects. So, it was quite a dynamic platform where one could publish posts with tags, reply to discussions, and see various author profiles.

While some features, such as advanced post filtering and multimedia support, were postponed due to time constraints, the blog page provided a strong foundation for future enhancements. My work on this component demonstrated the importance of creating intuitive and responsive interfaces that facilitate meaningful interactions among users.

Announcement Page Development

Another important area of my contribution was the announcements page. **This feature allows lecturers to easily post important updates, assignment deadlines and event notifications for their students.**

On this page I tried to make the interface clear and organised. This announcement page consisted of a text editor for the lecturer to type the message, and a display area where students could see the latest announcements. I used **React** to create reusable components that would format and display the announcements so that the interface would be functional and look nice. I also added a **filtering** option to sort the announcements by date or topic, which helped users find information when there was more than one update.



Graph 15: The announcements page

The development of the announcements page was a close team effort. I had to make sure that my front-end elements matched the design of the whole platform and interacted correctly with the **database**. In the end, the database structure and backend logic was mostly provided by Dawid, but I supported him by testing API endpoints and giving feedback on data flow.

User Profile Interface Development

Another feature I worked on extensively was the **user profile page**. This page allowed users to view their **personal information**, including their name, email address, and student ID, and provided the option to sign out securely. The design of the profile page emphasized **simplicity and intuitiveness**, ensuring that even users with limited technical expertise could navigate it easily.

Graph 16: The User Profile Interface

This feature was implemented by integrating **React with Firebase Firestore** to fetch and display user information dynamically. The **useParams hook** from React Router was used to retrieve the unique userId for each profile page, allowing the application to display personalized content for each user. For example, when a user navigates to another student's profile, the system dynamically fetches and displays that user's details from the database.

To ensure seamless operation, I implemented a **fallback mechanism**. If the system encountered an error fetching data from Firestore, it fell back to **mock profile data**. This ensured that the user interface would always show something meaningful in cases of network or server issues.

Key features of the user profile interface included:

- **Dynamic Data**: I used the **useEffect hook** to ensure that the user details load once the page opens. Thus, the information displayed to the user was always correct, and it pulled info like the user's name, email, and their student ID from the database.
- **Error Handling with Mock Data**: In case it failed to fetch from the database, the profile page then switches to mock data so that the interface remains fully functional and user-friendly even during adverse conditions.
- **Responsive Design**: Using **custom CSS and frameworks**, I made sure the profile page adapted seamlessly to different screen sizes, including desktops, tablets, and smartphones.
- **Secure Logout Option**: I added a sign-out button linked to the **/signout route**. This helps users log out faster, thus increasing user security.

One of the challenges I faced was to implement **robust error handling** while fetching data. For example, when the user data was missing or the database query failed, the interface had to function normally without broken elements. Having a structured fallback mechanism allowed me to resolve this pretty well.

The other main point is the **cooperation with Killian** regarding user roles and access levels. Information that was to be seen on the profile page includes, for example, if one is a lecturer or a student. Much of my cooperation with Killian has been about maintaining role-specific information coherently on the platform.

In all, the user profile page added value by finally giving users a page to **maintain and view their information** in. While the core functionality is implemented, there is definitely room for enhancements, such as uploading profile pictures directly on the interface or adding dashboards for activity tracking.

Challenges and Problem-Solving

Throughout the project, I had many problems that needed **creative problem-solving and collaboration**. Examples include smooth integration of the front-end with the back-end parts. For example, while developing the blog page, data were not loading from the database. These are often due to discrepancies between what was called at the front-end and what structure

was expected at the back-end. To fix this, I worked with **Dawid** on debugging API calls and aligning the format of the data.

Another challenge was dealing with the complexity of **real-time updates**, especially on the blog and announcement pages. While Firebase listeners did help with that, more testing was necessary to make them reliable. I have conducted several test scenarios to ensure that such updates worked in various situations, like simultaneous posts or comments by different users.

Chapter Summary

During SCRUM meetings, I served as Sprint leader and Minute man in meetings and understood the process of agile development more clearly. I always exchange opinions on the progress of the project, test and give feedback to other members on their works. In key phases of the project, I would think of the next development strategies and exchange the ideas with other members. I contributed some paragraphs of the public sections of this report as well. In conclusion, I feel that I've contributed enough to the frontend and the development process. I have always completed my tasks on time, made suggestions for this project, and helped other members of this team.

Chapter 5: Summary

This course has been a really enriching experience for me. I got to know about **Agile development** pretty well and how to deliver a functional product by working with the team. As an international student, this project meant something more because it was my first time collaborating with European classmates. Everybody in the team contributed their best effort, and together we created something we could all be proud of. This experience has deeply helped me understand the essence of teamwork and agile development, which I strongly believe will be helpful for my future career.

Overcoming Initial Challenges

During the initial stages of the course, communication was a big concern for me. Being an international student, I felt that my limited English proficiency might hinder the expression of my ideas or understanding of my teammates. This semester was my first semester in Ireland, and all previous studies were done in China. Though I had worked on group assignments with my classmates in China, this was my first time working with students from a different cultural and educational background

Luckily, my teammates were very understanding and supportive. Whenever I was struggling to understand something properly due to language barriers, they explained things clearly and used simple words so that I might catch up with the discussions easily. Gradually, I could involve myself more in discussions, which helped me raise my confidence and improve my English-speaking skills. Toward the end of this project, I felt considerably better at communicating in English, which I consider as the major achievement for myself.

Agile Development and SCRUM

Agile development methodologies during the project kept us systematic and flexible against the alteration in development processes. Thus, **SCRUM meetings** added to the very efficient fast solving of the problems through the sharing of one's progress, pointing out one's concerns, and collective mind storming for solution providers.

For example, in the beginning of this course, most of us had little idea about how to **use Git for version control**. So, one of our team members showed us how to use Git more effectively, teaching us how to handle branches and conflicts, how to merge code, and so on. Because of him, we learned how to handle version control in a team project, thus making collaboration smoother.

Tackling Challenges Together

The project wasn't without its challenges, but each obstacle became a chance to learn and grow. One of the biggest challenges was trying to get the scope and goals set for the project at an early stage. Because we were all new to the technologies involved, such as **React** and other libraries, it took some time to understand what we could actually achieve. It all came out through group discussions and brainstorming. Everyone combined to give a very clear roadmap, identified the project goals, user stories, and features to be implemented.

The agile approach allowed us to break down big tasks into smaller, more manageable ones. We worked in **two-week sprints**, assigning certain tasks to each member. Some members focused on research, while others started building template applications by following tutorials. This structured approach helped us transition smoothly from planning to actual development.

In the development process, there were many technical problems that came up, but with the help of teamwork and persistence, we were able to sort them out. An example was when integrating the front-end and back-end parts: there was an error in API calls. The front-end showed the parameters were incorrect, while the back-end developer alone could not find the problem. In the SCRUM session, front-end and back-end developers were working together, analyzing the code-when they found a mismatch in sending and receiving data. We successfully resolved the problem by modifying the mechanism of the API call, which showed the importance of collaboration in overcoming technical challenges.

Learning from Innovation and AI Tools

One of the most eye-openers was the use of AI tools in programming. I had never used such tools as GitHub Copilot or ChatGPT for coding before this course. They were recommended by my teammates, and I was really amazed at how they improved my efficiency. For example, Copilot helped me to perform repetitive code fast, while ChatGPT provided useful hints for debugging and solving some small issues.

In the process, however, I also learned the limits of these tools. Most of the AI code generated was unstable and hard to maintain, especially in a project with many interconnected files. There were instances where the AI suggested incorrect solutions that

could mislead. These weaknesses notwithstanding, I still found that AI tools could be useful learning aids. By asking the AI for references or explanations, I furthered my understanding of the code and concepts.

Team Dynamics and Issues Our team was generally easy to work with, but throughout, we had our ups and downs. Some team members seemed to wait until the last minute to do tasks, thus placing pressure on other team members. Others struggled to allocate time for the project since they were busy preparing for internship interviews or otherwise indisposed. These things sometimes brought up some friction, which we tried to overcome by talking openly. We discussed our concerns and adjusted our schedules to resolve the conflicts and move forward as a team.

Project Outcome

The outcome of our project turned out to be really satisfactory. The **CSConnect** platform included a blog page, an announcement page, and a user profile interface-all working in conjunction with one another. Especially, the blog page was the spot of the whole project, giving students a space to share ideas and collaborate.

Although there are still some bugs to fix, and we didn't complete the deployment due to budget and server limitations, the project laid a solid foundation for future development. We even discussed adding real-time chat functionality and improving the front-end design if we had more time and resources.

Personal Growth and Takeaways

I am proud of what we achieved as a team in retrospect. This project really taught me a lot about technical and interpersonal skills: now, I have so much more knowledge about Agile development, SCRUM, and Git version control compared to when I started. Te front-end and back-end development experience has grown my confidence to tackle similar projects in the future.

Personally, this project helped me learn better how to communicate and conduct myself in a multicultural working environment. From being initially afraid of speaking up because of hindrances in language, it has made me comfortable in trying to say what I want or discuss any topic with group participation.

Final Thoughts

This course has been an invaluable learning experience, and I believe it is a must for every computer science student. The course provides practical skills and prepares you for real-world challenges in software development. For me, it was a transformative journey which enhanced my technical knowledge, teamwork ability, and communication skills.

I would like to sincerely thank Kevin Casey, Simon Garrad, and all the lecturers and demonstrators who guided us throughout this course. Their support and encouragement played a significant role in our success.

We have had to work very hard in this project, but it's also been a supportive atmosphere for our team at all times. The successful completion of the CSConnect platform shows how much teamwork and dedication can work wonders, and I'm glad to have been a part of this project. It is a memory that shall stay with me long in my journey into the nook of computer science.

References

Anderson, D. J. (2010). Kanban: Successful Evolutionary Change for Your Technology Business. Blue Hole Press.

Balaji, S., & Murugaiyan, M. S. (2012). Waterfall vs. Agile: A comparative study on SDLC. International Journal of Information Technology and Business Management, 2(1), pp. 26–30.

Beck, K. (2000). Extreme Programming Explained: Embrace Change. Addison-Wesley.

Beck, K., et al. (2001). Manifesto for Agile Software Development. Available at: https://agilemanifesto.org

Beck, K., Beedle, M., van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., Grenning, J., Highsmith, J., Hunt, A., Jeffries, R., Kern, J., Marick, B., Martin, R. C., Mellor, S., Schwaber, K., Sutherland, J., & Thomas, D. (2001). Manifesto for Agile Software Development. Available at: http://agilemanifesto.org/

Cockburn, A. (2002). Agile Software Development. Addison-Wesley.

Cohn, M. (2005). Agile Estimating and Planning. Prentice Hall.

Cohn, M. (2006). User Stories Applied: For Agile Software Development. Addison-Wesley Professional.

Cohn, M. (2010). Succeeding with Agile: Software Development Using Scrum. Addison-Wesley Professional.

Fowler, M. (2001). The New Methodology. Available at: https://martinfowler.com/articles/newMethodology.html

Highsmith, J. (2001). Agile Software Development Ecosystems. Addison-Wesley.

Kniberg, H., & Skarin, M. (2010). Kanban and Scrum - Making the Most of Both. C4Media.

Larman, C., & Vodde, B. (2009). Scaling Lean and Agile Development. Addison-Wesley.

Leffingwell, D. (2011). Agile Software Requirements: Lean Requirements Practices for Teams, Programs, and the Enterprise. Addison-Wesley Professional.

Mahnič, V. (2012). Using Scrum in a Distributed Software Development: A Case Study. Informatica, 36(4), pp. 441-451.

Patton, J. (2014). User Story Mapping: Discover the Whole Story, Build the Right Product. O'Reilly Media.

Pereira, L., & Costa, J. (2022). Modern Web Development Practices. CodeMasters Press.

Rubin, K. S. (2013). Essential Scrum: A Practical Guide to the Most Popular Agile Process. Addison-Wesley Professional.

Schwaber, K. (2004). Agile Project Management with Scrum. Microsoft Press.

Schwaber, K., & Sutherland, J. (2020). The Scrum Guide. Available at: https://scrumguides.org

Sutherland, J., & Schwaber, K. (2017). The Scrum Guide. Scrum.org

React. (n.d.). *React – A JavaScript library for building user interfaces [*online]. React. Available at: https://reactjs.org/

Node.js. (n.d.). *Node.js®: JavaScript runtime [*online]. Available at: https://nodejs.org/

W3C (n.d.). *CSS: Cascading Style Sheets* [online]. Available at: https://www.w3.org/Style/CSS/

Firebase (n.d.). *Get to know Cloud Firestore* [online]. Available at: https://firebase.google.com/docs/firestore

Google Firebase. (n.d.). *Cloud Firestore Documentation*. Firebase [online]. Available at: https://firebase.google.com/docs/firestore

Firebase. (n.d.). *Firebase Authentication [*online]. Available at: https://firebase.google.com/products/auth

GitLab Inc. (n.d.). *GitLab* [online]. Available at: https://about.gitlab.com/

Beck, K., Beedle, M., Van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., … Thomas, D. (2001) Manifesto for Agile Software Development. Available at: https://agilemanifesto.org

Cohn, M., 2010. *Succeeding with Agile: Software Development Using Scrum*. Addison-Wesley.

Microsoft (2023) Microsoft Teams – Online Meeting and Collaboration Tool. Available at: https://www.microsoft.com/en-us/microsoft-teams

Miro (2023) Miro - Online Whiteboard for Visual Collaboration. Available at: https://miro.com

Schwaber, K. and Sutherland, J. (2020) The Scrum Guide: The Definitive Guide to Scrum: The Rules of the Game. Available at: https://scrumguides.org/

Appendix

Team Meeting Minutes:

Team 14 Meeting 2 Minutes
Present: Kian, Emmanuel, Killian, Dawid, Songyan , Danila
Absent: Brian

Proposal:
Based around a Discord style app. With each module getting it's own page consisting of a blog page and discussion forum.
Accounts to require university email to register.

User Stories:
Consolidated all user stories into one document. Each person read out their user stories. Killian, Dawid, Songyan were tasked with cutting down on duplicate stories and sorting them into the categories of Admin, Lecturer, CS Rep and a general user role. The team then created the story map, from these user stories.

Miro:
Kian created the Miro board and invited the team. Backlog was created on the Miro board

Sprint Planning:
Tasks to be estimated in terms of complexity. Kian, Emmanuel and Danila sorted the proposed features into could do, should do, must do. Emmanuel chosen as the first sprint leader. Tasks were voted upon in terms of complexity and assigned to each other.

Midweek-Meeting 2 Minutes
Present: Kian, Emmanuel, Killian, Dawid, Songyan , Danila, Brian

Today we went over the work everyone has done so far. Such as the chat and profile features seen earlier in the teams.

We also discussed problems people have been having with fire base and react. We brainstormed some solutions to these issues.

We also taught of rough ideas for the next sprint, work that needs to be nailed down such as ui design and colouring.

Overall, very good meeting today.

Meeting 3 Minutes
Present: Kian, Emmanuel, Killian, Dawid, Songyan , Danila, Brian
Absent:

Re-cap :
Team re-capped on last Wednesday's and Monday's meetings.

Backlog :
Team updated the backlog based on new tasks and completed ones. Team members took tasks that were similar to their sample tasks from last week. Team re-estimated the complexity of each task based on team feedback.

Gitlab :
Danila setup the gitlab for the team and invited everyone.
Kian setup the basic project on gitlab with node js, react and firebase.
Each team member sets up their own branch for their code, team will merge code later when agreed.

Project name and technologies:
Team compiled name ideas and settled on CSConnect.
Technologies to be used: Node.js, React, Firebase, CSS, Javascript. Gitlab

Meeting 4 Minutes
Present: Kian, Emmanuel, Killian, Dawid, Songyan , Danila, Brian
Absent:

Sprint Re-cap :
Kian made a chat feature, working on file upload. Supports multiple accounts
Danila made the login in and sign up pages.
Songyan made the announcement page sample, along with the user profile page.
Dawid made the firebase database for the project.
Emmanuel made the UI concepts and wireframes. Was also the sprint leader.
Brian made the search feature for the app.
Killian is working the home page.

Sprint Retrospective :
We could have had better communication outside of meetings .
We could have made our goals for the project a bit more clear.
We could have been a bit more clear on the style of the app and the overall ui/ux.

As a team we mostly got through our sprint. Everyone stuck to their tasks when developing the project so far.
We have been having multiple meetings each week.

New Sprint Leader :
Kian is the new sprint leader. Danila will be taking the minutes.

Sprint planning :
Team got an average opinion of the difficulty of the tasks for the next sprint using Planning Poker.
Distributed tasks based on difficulty. Allocation is on Miro.
Emmanuel starts research paper on css styles for app. Making sure every page looks similar.

Problems/Solutions:
Discussed what problems anyone is having and came up with fixes/work-arounds.

1. Found reason for some ‘npm’ .jsx files not running
2. Helped connect firebase to .jsx files where connection is required
3. Some files take longer than anticipated
4. Difficulty in designing UI/UX wireframe due to lack of knowledge – Research on competitor UI.

- Coming to an agreement for the apps design.

5. Connect the user profile, announcement page, and blog page to the database

Miro:
Updated backlog for next sprint.
Updated story map.

Week 5 minutes:
Present: Kian, Songyan, Dawid
Discussed the work everyone has completed over the week. This included issues that have come up with certain features.
We added a couple potential features to the miro board, and clarified the goals for others.
We examined how different everyones styling for their apps is so far as we wait for the research paper to be finished.
We looked to other programs such as teams and discord for inspiration on styling.

Midweek 5 Minutes:
Present: Kian, Emmanuel, Killian, Dawid, Songyan , Danila, Brian
discussed progress on current sprint
discussed any problems and tried to answer questions anyone had
caught up the people that missed last meeting.
Week 6 Minutes:
Present: Kian, Emmanuel, Killian, Dawid, Songyan , Danila, Brian
Sprint leader changed from Kian to Danila.
Minutes changed from Danila to Brian.

Sprint recap:
Kian most of the chat feature is complete other than linking to database. (waiting for merge)
Danila finished email verification for mumail.ie only and password (length +8 length, min 1 letter, min 1 digit)
Songyan finished blog page, issue connect database (waiting for merge), issue on uploading photo.
Emmanuel finished research paper.
Killian finished homepage
Brian finished user list and user search, issue on professor with js (used css)
Dawid finished js code for database connection in login/signup

Retrospect:
Improve checking teams regularly and communication, improve showing off work on teams
Research paper requires more resources for css design.
Hope for the best with merging.

Sprint Planning:
Team got an average opinion of the difficulty of the tasks for the next sprint using Planning Poker.
Distributed tasks based on difficulty. Allocation is on Miro.
Team looked at UI mockup and research paper, agreed on final css style for overall app.

Week 7 Minutes:
Present: Brian, Kian, Emmanuel, Killian, Dawid, Songyan , Danila
Absent:
Sprint Recap: Discussed and showcased the work everyone has completed over the week and talked about any issue that had arise over the weekend. This sprint was graded and was graded full marks. No new task has been handed out as this week has been very busy for team members

New Sprint leader/ minute man:
Sprint leader changed from Danila to Brian
Minute man changed from Brian to Dawid

Problems & Solutions:
1)
Problem: Npm wasn't running, would give "npm" not found error.
Solution: Found solution to why npm error kept popping up

2)
Problem: Merging problems
Solution: No solutions yet

Week 8 Minutes (Monday)
Present: Kian, Emmanuel, Killian, Dawid, Songyan, Danila, Brian
Absent:
Re-cap:
We discussed the work everyone has completed over the week.
Backlog:
We continue working on the features in the backlog.
Difficulties:
We addressed and talked about the problems that have occurred during the week.
Merging:
We discussed how merging can affect out development process and the setbacks it may cause.

Week 8 Minutes (Wednesday)
Present: Kian, Emmanuel, Killian, Dawid, Songyan, Danila, Brian
Absent:
New sprint leader / minute man:
Sprint leader changed from Brian to Dawid
Minute man changed from Dawid to Killian
Re-cap:

We discussed the work everyone has completed over the week.
Backlog:
We continue working on the features in the backlog. Killian started working on the report.
Difficulties:
Addressed any concerns that occurred during the week (css errors, firebase errors, changes in page styling).
We have experienced errors when merging branches and we are trying to solve them the following week.
Page styling:
We talked about the fonts and colours that the website should be styled in.

Week 9 Minutes:
(Monday)
Present: Kian, Emmanuel, Killian, Danila, Dawid
Absent: Brian and Songyan
Sprint Re-cap:
We discussed the work everyone has done for the week by showing the work we have done so far.
Kian was able to merge the homepage, blog page, chat page together for testing
Dawid finished connecting everyone's page to the database
Killian finished the homepage and connecting to the database, now working on the report
Danila finished connecting the login and password to the database
Emmanuel updated the Navigation page
Problems faced:
During the meeting when discussing the tasks everyone had done , we faced a lot of problems especially with the beginning of merging everyone's code together , which caused a bit of a crash in the system but thanks to Kian we got some of the problems fixed and hopefully we can fix the other problems in the next meeting on Wednesday.

Week 9 Minutes:
(Wednesday):
Present: Kian, Emmanuel, Killian, Danila, Brian, Songyan and Dawid
Absent:
New sprint leader / minute man:
Sprint leader changed from Dawid to Killian
Minute man changed from Danila to Songyan
Recap:
Going through what everyone has done so far:

- Kian : Merged eveything together, just waiting for a few things to be uploaded to gitlab
- Danila: Got finished connecting the login page to the database and finishing to ale sure that the user that logs in , to stay logged in all of the pages
- Emmanuel: Pushed his branch onto gitlab that represents the Page Layout Style for our project
- Brian : Finished uploading the Search bar Function and linking it up to the database
- Dawid: Finished connecting most pages for our project to the database using Firebase, still going through , making sure everything is connected properly

- Songyan: Finshed getting the Blog and Modules Page
- Killian: Finshed uploading the Homepage and connecting it to the database , now starting the Group Report

Retrospect:

- I think we need to improve looking at teams a little bit more
- Merging is on point just a few little problems but nothing to severe
- Lastly being on time can improve

Week 10 Minutes:
(Monday)
Present: Kian, Emmanuel, Dawid, Songyan, Brian
Absent: Danila, Killian
Sprint Re-cap:
Kian: Added a search function to every page, ensuring each page has this functionality.
Emmanuel: Implemented global CSS styling across the application.
Dawid: Worked on color schemes and implemented delete functions.
Songyan: Connected the user profile to each blog post on the blog page.
Brian: doing the search page
Problems faced:
Encountered issues with the chat function in each module.
The delete function in the messaging feature is not working.

Week 10 Minutes:
(Wednesday):
Present: Kian, Emmanuel, Killian, Danila, Brian, Songyan and Dawid
Absent:
New sprint leader / minute man:
Sprint leader changed from Killian to Songyan
Minute man changed from Songyan to Emmanuel
Recap:
Going through what everyone has done so far:

- Kian: Updated the chat function to the login, merged the search feature, adjusted the CSS, and merged the user profile.
- Danila: Managed the sign-up process for the current account and established the connection between the current account and the web.
- Emmanuel: Fixed the CSS style on each page.
- Brian: Worked on the search page.
- Dawid: Linked the database to the user profile, created different colors for different users, and implemented the delete function.
- Songyan: Completed the code connection between the blog and user profile.
- Killian: Wrote the project report.

Retrospect:
We should start working on our documentation.
We need to communicate more.

Difficulties:

Figuring out the referential relationships in the CSS styles.

Team 14 Project Proposal:

CS Social Website

Features
Chat function – pages?, create argument on whether chat function should be just groupchats or student to student.
Blog Page – allowed to upload videos, comments?
Different permissions for CS REP and lecturers etc.
User Stories of features you'd like.

Project
A page where users can make posts, website like discord, twitter or Facebook. CS reps and lecturers should have different permissions, such as pinning their posts to the top of the page.

Extra features
Blog posts should be able to include images and videos
Chat functionality for the website
Comments to posts

User Stories
Login and Registration
As a user I want to be able to register to the website so that I can access its features.
As a user I want to be able to log in to the website so that I can access its features.

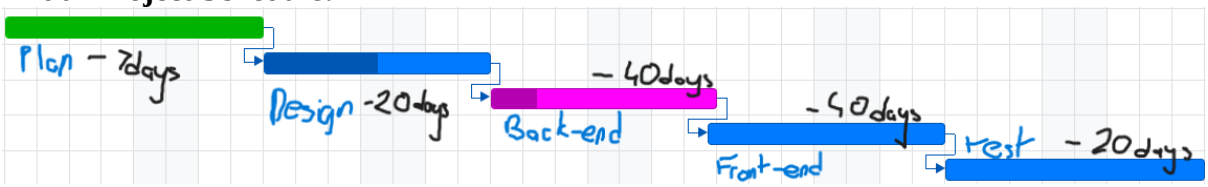
Post creation/functionality
As a user I want to be able to fill a post form, so that I can create a post.
As a user I want to be able to click an "edit" button, so that I can edit contents of a post that I created.
As a user I want to be able to click a "delete" button, so that I can delete a post that I created.
As an administrator, I want to create a class groups, so students and lecturers can communicate together.
As a user, I want to be able to send messages to a group, so I can ask for help or collaborate individually.
As a student, I want to be notified when a new post or announcement is made, so I can stay updated on new information.
As a student, I want to be able to receive get comments on my posts, so I can improve in the future.
As a CS representative, I want to be able to post announcements so I can inform students about department events and deadlines.
As a CS representative, I want to be able to delete inappropriate comments, so I can maintain a positive online environment.
As a lecturer, I want to be able to post course announcements on the main group, notify students about any upcoming events and deadlines.
As a user, I want to be able to comment on blog posts, so l can answer questions and give back feedback.

As a user, I want to receive notifications when other users comment on my posts, so l can engage with them.
As a lecturer I want to be able to pin my announcement so that it is visible at the top of the page.

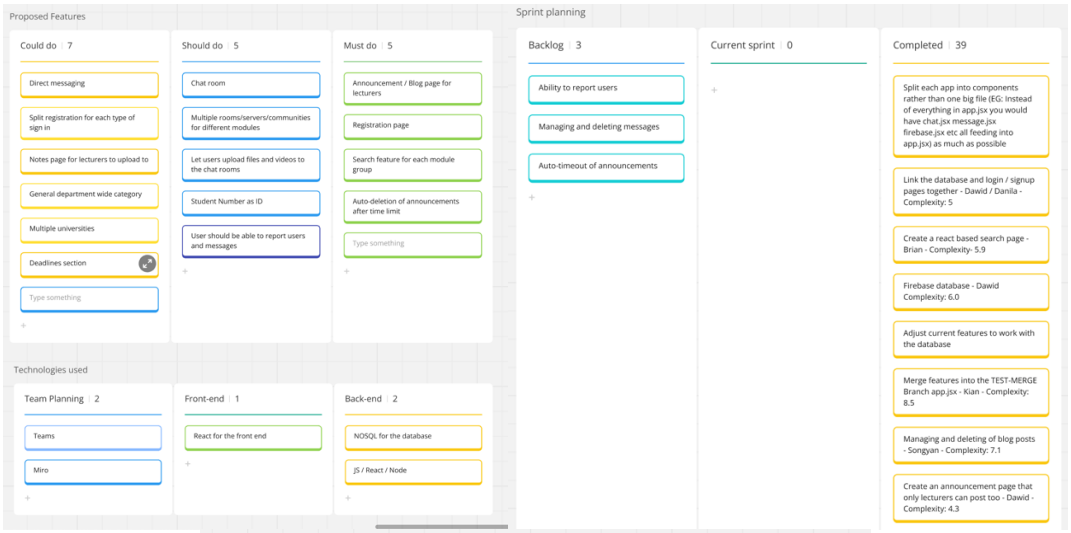
User scenarios
Matt is a lecturer at a university. He's in the middle of a workday giving lectures to different classes.
Matt has been informed of a change in the venue availability and he needs to inform the class.
He opens the CS Social Website on his laptop. After logging in he sees the newest posts. To inform the class of the change he presses the "create a post" button, fills the form and finalizes the creation of the post.
To make sure that the post will be easily visible to all users he proceeds to pin the post to the top of the page.
About an hour after making his post, Matt gets notified by the administration of another change and the original venue can be used.
To inform the class, he opens the CS Social Website and adds a comment to his post informing of the situation.
After finishing his lecture Matt decides to unpin his post, so that it no longer appears at the top of the page.

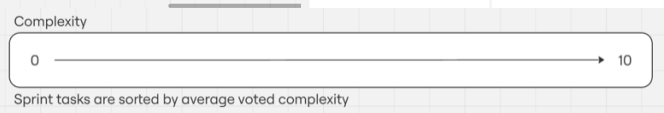
Name: John, a third-year Computer Science student
Scenario:
John has been assigned to a group project with his classmates and needs to discuss the project details. She logs into the Social Website to communicate with his team and ask her lecturer questions.
John opens the website and navigates to the chat function.
He creates a group chat with her three other classmates.
In the group chat, John shares various of links to online resources and uploads a project proposal for his group to look at.
After discussing with his group, John goes to the blog page, where the lecturer has posted more coursework materials for help.
John comments on the lecturer's post to ask for clarification on the assignment that was given to get a better understanding of what's asked.

Initial Project Schedule:



Miro Board:





Create user list, spliting by lecturer and student - Brian - Complexity: 5.6

Start on global css file for the project - Emmanuel - Complexity: 6.9

Mock UI for all the pages - Emmanuel - Complexity: 5.7

Start on the home page - Killian - Complexity 6.3

Student number during reigstration - Danila - Complexity: 3.5

Link search to the database - Brian. - Complexity : 5.2

Create a sample database in nosql. Tables: Users, Modules - Dawid - Complexity: 6.2

Continue work on the homepage - Killian - Complexity: 6.3

Create a sample registration page to test - Danila - Complexity 5.2

File uploading - Kian - Complexity: 4.5

Create main chat function - Kian - Complexity: 7.3

Make the search a popup - Brian. - Complexity : 5.2

Create a user profile(name, student id, student email, major, photo) - Songyan

Create different sections for different modules - Songyan - Complexity: 5.2

Create a sample announcement pages that times out messages after a certrain amount of time. Songyan - Complexity: 6.7

Link the chat function to the database - Kian - Complexity: 3.8

Create a sample search page. - Brian - 4.3

Port js to jsx, - Kian - Complexity: 3.8

Use react to create announcement page. Songyan

Add studnent number as user another ID - Dawid - Complexity: 4

Create a sample chat feature to test the plausability (both dms and group) - Kian - Complexity: 7.3

Restricing the log-in and sign-up to specific emails (mumail, mu.ie) and password verification (minimum length, 1 cap 1 digit)- Dawid / Danila - Complexity: 3.9

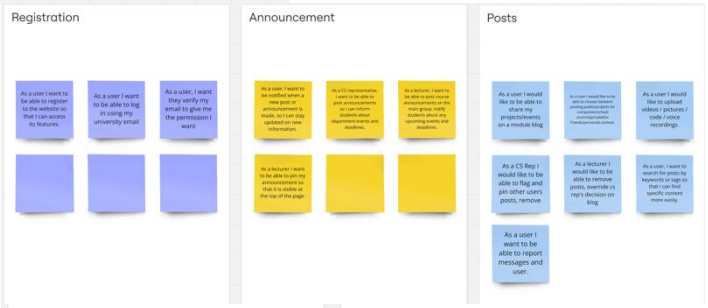
Create a sample database in sql. Tables: Users, Modules - Killian. - Complexity 6.5

Connect the user profile to the database - Dawid - Complexity: 6

Calendar feature -Songyan

Work on the blog page - Songyan - Complexity - Complexity: 5.9

Create links to user pages on posts - Dawid

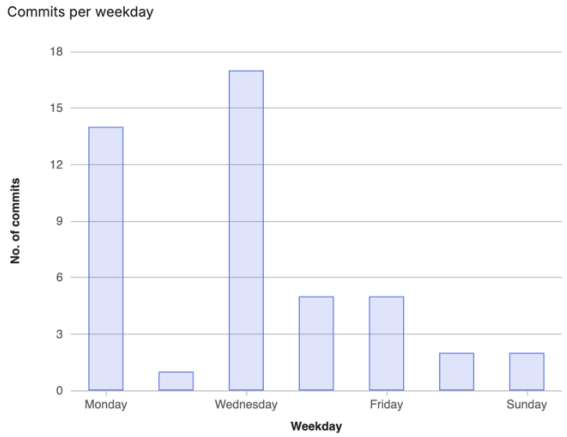


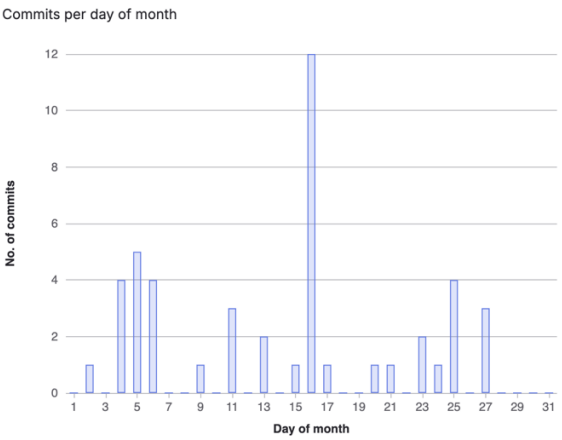
Create a sign out button connected to database- Danila

Add different colors to posts for users of different access levels - Dawid

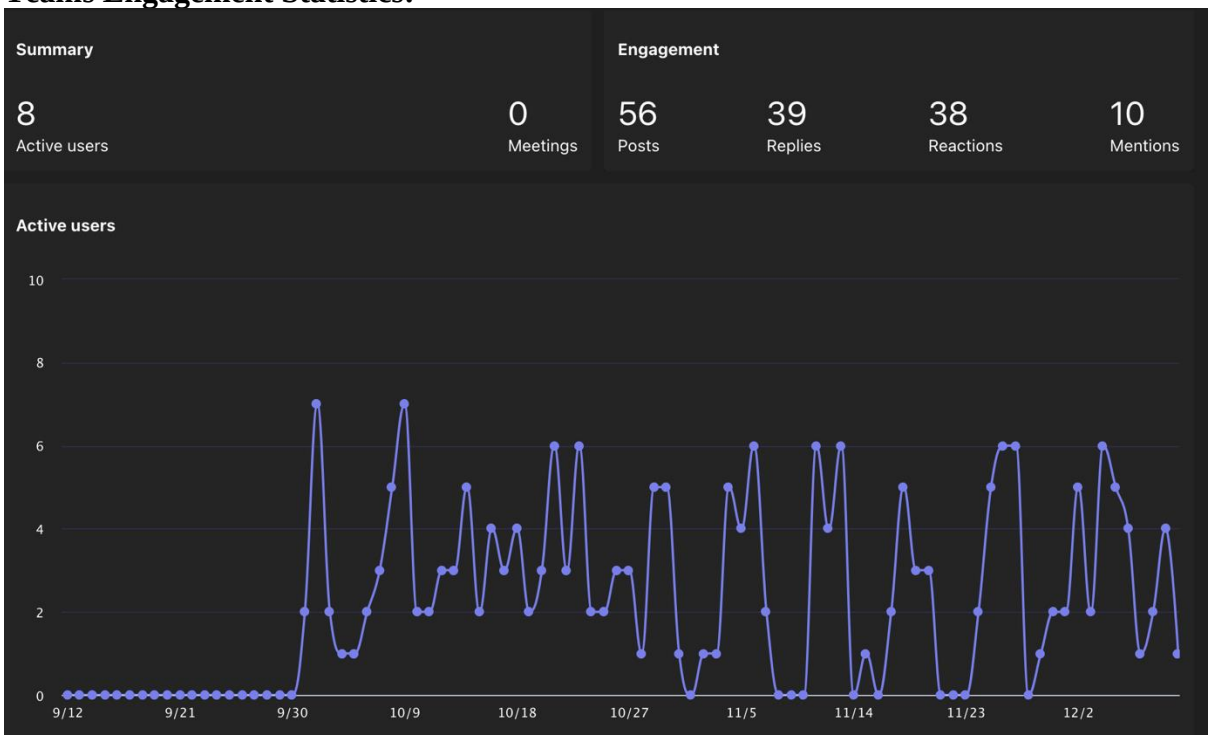
Connection between the blog and user profile -Songyan

Git Lab Analytics





Teams Engagement Statistics:



CSS Research Paper:

Research Paper on UI/UX Design for a Student-Focused Website

Abstract

The purpose of this work is to create an enjoyable and useful website for Computer Science students that will combine the opportunities of LinkedIn and Discord. The job of a UI/UX designer is about making the product more usable by carefully choosing colours, arrangement of elements and ways of their interactions. This paper focuses on design consideration of Computer Science students especially in aspects such as colour psychology, structuring of the interface and information about the target group.

Little is known about the target consumer group that the commercials are depicting. Computer Science students need a tool that combines teaching and learning materials, communication tools, and a way to showcase their work. Works aimed at studying student engagements and their activity levels show that students use several sites at once: LinkedIn for contact and Discord for communication and collaboration. Combining these functionalities in a single site can benefit them especially in sharing of content related to specific modules and interacting throughout the social network.

Based on this research, the primary design goals should be:

1. Accessibility: Making sure student with different levels of computer literacy are comfortable with the platform.

2. Engagement: Designing of the look and feel of the interface while promoting relational and achievement objectives.

3. Resource Consolidation: Allows the learners to compile their study material in an efficient manner.

The Subject of Colour Psychology and Its Implications for the Field of Design

1. They play an important role of shaping user experience since they trigger unique emotions and behaviours patterns. Knowledge of colour psychology is crucial in creating an interface that would appeal to users

   Primary Colours: Blue and Green

   - **Blue** is known to be trustworthy, stable and intelligent and therefore is an excellent primary colour for our design

   o Green is perfect when used together with blue that is why it becomes easy to achieve the stress-free atmosphere.t. Green is also associated with growth, and we have included knowledge since the goal of the educational platform.

   Accent Colours: Orange or Yellow

   - Orange: It can also be used in with the concerns related to buttons or other notifications which deserve the special attention. Orange is bright and spirited and can command notice without becoming obtrusive and this will help to sustain user interest without being disruptive.

   o **Yellow** is suitable for the use in the alerts, messages or call-to-action items. It can help motivate users by illuminating messages and brighten the experience or bring a friendly touch.

   Background and Neutral Colours

   - Avoiding the use of bright colours for backgrounds is important so that the contents of the website take central stage instead of the interface.This lowers the amount of effort required in the user's working memory when in content heavy platforms.

Design Features and Layout2

Having two main use cases – content sharing which is like LinkedIn, and the chat function which shares some similarities to Discord, it is an opportunity to have an opportunity to work on an integrated UI with clear navigation.

LinkedIn-Style Modules and Portfolio Sections

- Modules will enable the students to upload related multimedia contents for their courses. The organization of content is in the grid view for convenience and 2) and each module has the cards on the homepage.
- Each of the above module cards can also have a subject name, thumbnail and icon that can help users to interact. Tooltips are a way to tell the user what a certain button or icon is for which can help in a better understanding of this application.

Discord-Style Chat Functionality

- A side or bottom bar pop-up email chat window is useful to allow users switch between content and chat without any interruption.
- Allowing students to create chat rooms or join rooms of a certain subject provides a sense of community and increases activity of other and group work.

User Profile Customization

- Options like visibility of users, pictures, colours used, and posted status, this will help students to have an enhanced identity on the social media.

Advantages of the Proposed Design

Enhanced Focus and Engagement: By using colours strategically, the platform encourages focus and a calm learning environment.

1. **Clear and Intuitive Navigation**: With a minimalist design and interactive tooltips, users can navigate with ease.
2. **Sense of Community**: Integrating chat features with a professional portfolio section encourages students to support one another, share ideas, and form study groups.
3. **Resource Management**: The modular design helps students keep their content organized, improving access to personal notes, videos, and other learning materials.

Disadvantages and Considerations

1. **Balancing Professionalism and Casual Interaction**: Finding the right balance between the professional vibe of LinkedIn and the casual tone of Discord can be challenging. Overemphasizing one aspect could deter users looking for a different experience.
2. **Potential Overwhelm with Too Many Features**: A dual-functional website could risk overwhelming first-time users. To mitigate this, our design should prioritize simplicity, with optional tutorials or tooltips for onboarding.

Conclusion

Based on this research, it is proposed that the site design will incorporate a measure of professionalism whilst also remaining accessible to the community through proper colour contrast and an adequate structure. Anticipated visual themes include primary colours, interactive accents, and the Linkedin-style content-blocks set against the Discord-inspired chat interface. This project aims at addressing the needs of Computer Science students and provide a platform that will enable students meet, learn and develop, all manageable within the platform.

References

- BeanMachine. (2023). *The Psychology of Color in UI Design: How to Influence User Behavior*. Retrieved from https://beanmachine.dev/psychology-of-color-in-ui-design/
- Eagle. (2023). *Color Psychology in Design: How Colours Impact User Perception and Experience*. Retrieved from https://en.eagle.cool/blog/post/how-color-psychology-impact-user-perception-experience
- Interaction Design Foundation. (2024). *The 10 Most Inspirational UI Examples in 2024*. Retrieved from https://www.interaction-design.org/literature/article/ui-design-examples
- ReloadUX. (2023). *The Psychology of Color in UI Design* Retrieved from https://reloadux.com/blog/the-psychology-of-color-in-ui-design/