**COS 301 Semester Test 1**

**Scott Bebington**

**U21546216**

**Question 1**

1. Being able to filter content.
   1. By clicking on posts, Instagram automatically generates a “Filter” for what you may like.
   2. Going to the profile page to block sensitive content.
   3. Being able to block accounts to not show their content.
2. Being able to share content.
   1. Using hashtags to link in themes and trends.
   2. Being able to post photos, videos, gifs etc…
   3. Adding captions to content being posted.
3. Live sharing a user’s experience through IGTV and live stories.
   1. Being able to show what a user is doing if they chose.
   2. The ability for the live to be liked.
   3. A comments section for watchers to comment if they are allowed to.
4. Direct communication with another user
   1. Being able to video call/voice call/message a user.
   2. Commenting on posts
   3. Liking posts
5. Being able to shop for items.
   1. View items on sale.
   2. Share items via story or direct message.
   3. Being able to go to the website that the items are being sold on.
6. The ability to search for specific things.
   1. Searching for a hashtag.
   2. Searching for an account.
   3. Searching for posts taken at a specific place.

**Question 3**

1. Usability: For a service such as Instagram top work it has to be easy to use for people of all ages and technological backgrounds. This is the case for Instagram as there isn’t much going on per page, making it easy for the user to navigate through everything.
2. Availability: Instagram is available on any device in the form of an app or website making it easy to access at any time.
3. Scalability: Instagram most likely has millions of concurrent users yet the performance on the app remains constant for each user and never slows down because someone has to wait for a server to respond.
4. Security: Instagram has great privacy and data protection built into the app so you don’t have to worry about data breaches. It also uses very strong user authentication and authorisation ensuring that nobody can do anything they aren’t supposed to do.
5. Maintainability: Instagram often has UI updates and functional updates that don’t affect the user apart from the occasional software update that can be done at any time.

**Question 4**

Usability

1. Multiple sources: The Instagram service should be available from more than 1 place, such as a phone app or a web service.
2. Simple design: Having few tasks per page may make it easier to navigate an app and use it properly.
3. Visually appealing: Having a visually appealing app may make it easier to navigate.

Availability

1. Prevent errors: Ensure data redundancy so that is anything fails on the DB or anywhere else, there is a copy of the data. Developers should also thoroughly test their code before pushing to production.
2. Detect errors: Exception handling should be used, deadlock prevention should be used, data integrity should be maintained.
3. Recover from errors: If anything fails it should be fixed as soon as possible, rollback to an earlier stable version if necessary.

Scalability

1. Vertical Scaling: If there is too much traffic on a server it may be a good idea to upgrade the components on a server to ensure faster process time or more memory etc.
2. Horizontal Scaling: If there is too much traffic on a server it may be a good idea to add another one or more to handle the increased load.
3. Manage resources: if multiple servers exist, there should be a process that directs incoming traffic to the correct server to ensure that no 1 server is being overloaded.

Security

1. Deter Attacks: By limiting the user to only performing certain tasks, providing authentication, and checking authorisation, Instagram is less likely to be attacked.
2. Detect Attacks: Check logs for any suspicious data. Do message checking before being uploaded to the server and DB.
3. Recover from Attacks: After an attack the developers should find out where the attack came from and make sure it cant happen again.

Maintainability

1. Lowly coupled code: Make sure code isn’t highly coupled so that if a component needs to be updated, nothing else will need to be updated in order to work.
2. Documentation: well documented code may make it easier to update if a new programmer looks at it.
3. Code readability: Make sure the code is able to be understood and simple enough to change if need be.

**Question 5**

A challenge our team came across was mainly time management and communication, we had an issue somewhat early on where a member of another team was assigned to function that was supposed to be done by a member of the UI engineers’ team by our project manager, this was because the UI engineers had not started working due to other commitments. This caused problems as none of the member of our UI team were notified and when the function was being worked on by the member of the UI team, when it came to be merged with the main branch it was discovered that it had already been done. To avoid conflicts like this in the future I mentioned to the project manager that they should communicate any changes to all teams and to only outsource work if a member of a team physically cannot complete the work or they need help.

**Question 6**

1. Goal setting

Software Engineering: Software Engineering often focuses on long term goals that align on the project as a whole, such as specific deadlines that need to be met, i.e. Setting up the database, Frontend being done, Edge functions fully complete. Setting up infrastructure by making sure the work environments, physical and per device, is able to be used correctly.

Software development: Software development more focuses on short term goals for completing smaller tasks such as when a component of a webpage needs to be complete, when the database rules need to be done by or when populating a table or data needs to be complete.

1. Problem solving

Software Engineering: Often involved lots of going back and forth between the client, not so much in the mini project, more so in the capstone project. Deciding on tech stacks to use, conflicts between members of the team

Software development: this focusses more on solved code related bugs, getting data to work in a specific way, making sure all the functional requirements are met.

1. Quality checking

Software Engineering: This more focusses on making sure the client is happy with the end result of the product, in our case making sure that demo 2 goes well and that all the needs for the project have gone well, making sure the users are happy by gathering user feedback, comparing our mini project with its actual social media to make sure it looks as similar as possible and has all the requirements.

Software Development: This focuses more on making sure the code that is being written runs correctly by using tests set forth by the devops guys, making sure data that is passed back and forth between the database is secure and validated. Testing all edge cases to make sure nothing breaks down in a production environment (we had a few instances of using *“* ’*”* in text which broke our production website as it wasn’t supported)

1. Project Scope

Software Engineering: This involves getting a broad idea of what exactly needs to happen for the project to work, what are the functional requirements, for us it was what does the user need to be able to do on the website. What technologies will be used to create a project, this may include a framework such as react to be used for the frontend, supabase to be used for the backend and vitest to be used to do frontend testing and edge functions to “Host” out API.

Software Development: This involves more fine-tuned scopes. For example, we are using next-ui as our component library. For the UI engineers our scope would be per page and deciding what components from next-ui would be a viable option to make our social media clone the same as the actual social media site.

1. Resource usage

Software Engineering: when making sure the resources are managed correctly, this is about making sure you have the right people working to their strength on the project and making sure the budget is being properly monitored and adhered to.

Software Development: This is making sure functions aren’t to CPU or GPU intensive, making the code more efficient. Making sure you aren’t spending to much time on any given function or component.

**Question 7**

Software architecture is the process of visualising creative work with scientific knowledge to design a system that meets the required properties the client wants. this worked well in our project as we saw the requirements of the project and used people strengths to design and be creative as well as the members strengths when needing to implement the systems. this means we can build a more robust and beautiful looking system.

**Question 8**

Sprint planning: this involves the members of an organisation to discuss what will be and what wont be done during a sprint.

Meetings: Often software companies have a daily standup meeting which each member will briefly say what they had done on the previous workday and what they plan to do. We are using a discord daily-bot to streamline the process.

Designing: This is when new features need to be discussed and designed to fit into the project

Implementing: This is when new features are actually completed.

Code testing: This involves testing the code for a new feature to ensure it works in the correct way and under all cases.

Code review: This is the process of generally getting more senior developers to go through code and make sure it meets all requirements set out.

Deployment: assuming your code makes it through code review it will then be deployed to a deployment server.

Reflection: After the sprint is finished a meeting should take place to discuss what happened in the meeting, how everyone feels, what can be done better etc. if it wasn’t already brought up sometime during the sprint.

**Question 9**

We decided to go with the agile methodology for our mini project as we wanted to promote short lived branches when designing our project. This means that for each process and component we make, each one is individually planned as well as planned our in the broad spectrum. We thought this would more effectively that planning everything out one and then jumping straight into development as more problems could be made and more redesigns would have to happen if issues occurred. Another reason we went with this methodology is because after each component is complete it doesn’t have to be looked at again until it has to be merged with every other component, with waterfall, every component would have to be looked at and compared to others.

**Question 10**

Question 11

Wireframes are a very basic way of showing the process that a user will go through when using an application.

Mock-ups give the client a preview of what the final product that will be served by the developers.

It helps the process by showcasing how the system will be available, reliable, secure and scalable.

**Question 12**

Unit testing: this involves testing each individual component separately to ensure that it works correctly by itself. Eg. Making sure a card containing an image and some text appears correctly.

Integration Testing: This involves testing components as a group to ensure all of them work together. Eg. Making sure a page works correctly with a searchbar and a few users posts.

End-to-end testing: this involves testing the entire system as a whole. This means going through the system as a user would and making sure everything works.