1. What design patterns are you going to use? What is the reason for using it?

We will be using Model View Controller design pattern because the use of MVC allows us to divide the project into small sections and each component of the program can be designed, tested, and operated independently. This way, if there are any changes in the design or implementation, the whole product won’t be affected, fixing the small parts and combining them would be more efficient.

1. What technologies / frameworks are you going to use to implement the solution? (20)

For question 2 we would include implementation options like java, .NET, Eclipse, Visual Studio,

CVS/Subversion and TIBCO giving the client the advantage and disadvantage, cost of the project and availability of resources for each of them. We would also add encryption, decryption and session management which are security related tools.

1. What development methodology are you planning to use and why? (20)

We are going to use Agile development methodology because ​​it​’s ​easier to​ ​change​ ​and​ ​allows​ ​much​ ​more flexibility​ ​than​ ​traditional​ ​methods. Clients can provide feedback easily & any kind of changes can be made according to it. So it makes successively refine the software system without affecting the budget & deliver it on time.

1. Discuss testing methodology to make sure all functional requirements are met. (20)

We will ensure our admin portal meets all functional requirements through unit testing. Unit testing involves the isolated testing of each module in a program. By using a batch of test instructions or data, we can easily and regularly test our portal modules to ensure they meet all functional requirements. Our portal would have the functional requirements: it operates solely on the company network; admins only have access to links based on their roles; some links are available to all admins; and that our portal will be able to be interfaced with a third-party program. Our design approach is heavily modular, so it follows that testing of each individual component as it is built will make it easier to integrate all the pieces in the later stages of development. If each individual module of the design functions when separated, the program should function when the modules are integrated. If a module of the design does not meet functional requirements, then it can quickly be rectified to prevent issues when the modules are integrated. Our MVC architecture allows for the easy testing of each component. The View component can be tested using a correct login and an invalid login to ensure that access is only being granted to verified admins. The Controller component can be tested by passing valid roles and ensuring the correct links are returned, and if an invalid role is passed, the program will notify the user. The Model component can be tested with admin types ensuring that higher level admins are able to modify permissions while lower level admins would not be able to access these functions. For the integration of these models, integration testing can be done towards the end of the project to ensure that the program functions when the modules are integrated.

1. How will you address the changes in requirements and resources when you are half way done with implementation? What areas will be impacted with changes? (20)

As we have opted for an agile design methodology, it is likely that our requirements will have already changed by the time the project is halfway completed. The involvement of our client during the development process of our software will best allow us to adapt to our client’s needs following incremental reviews of the software. While the functional requirements will be unlikely to drastically change during development, the client may request additional non-functional requirements to improve workflow or general ease of use within the program. This could involve added functionality for high level admins over lower level admins, or a custom designed user interface to improve usability. Our agile design methodology, and consequently modular program, will allow for the easy addition of non-functional requirement modules. In comparison to waterfall design methodology, agile and modular development is beneficial. A waterfall design methodology would not be easily able to adapt to changes in the program requirements. It would generally require a complete overhaul of the partially developed program with little salvageable code and lead to wasted labour. Our modular design is not affected as badly by changes in program requirements. By the halfway completion point of our implementation, many of our modules will have been completed and unit tested, so we will begin to move from a predominantly unit testing environment to an integration testing environment to make the integration process of all the modules smoother.