SLIDES 11-17 DESIGN

Design considerations for a cloud-based solution

Slide 1. ACCESSIBILITY (designing for).

From Jamsa:
‘Depending on the processing that a system performs, a designer may need to create an interface that maximises user access or may have to lock down the system and control which users can access specific features. For a public solution – eg consumer website – maximising user access not only makes marketing sense, but is required by law (disability access). In contrast, for a secure site, controlling user access can range from ensuring the security of the login process to some type of biometric user authentication.

Slide 2. AUDIT (designing for).

Jamsa:
‘As you design a cloud-based solution, you must identify critical processing points at which you will want to *place an internal control to confirm that the solution’s processing is correct* and *free from outside manipulation.*
You may design such controls to be *active*, meaning the *code may generate a processing exception* should unexpected results occur, or *passive*, possibly *logging events to a file* or saving *snapshots* of data’

Slide 3. AVAILABILITY (designing for).
Jamsa:
‘As part of their SLAs, most cloud providers guarantee system availability, normally specifying a percentage of uptime (eg 99.99%) ... it is important that you identify your system’s uptime requirement and then, likely through the use of redundant colocated servers, design a solution that meets your needs.’

Slide 4. BACKUP (designing for).

‘Loss of user data should not occur. ... you must consider not only ways to backup your data and databases, but also the impact of each method on your system availability should you need to bring down the system to restore a backup.
‘Designing redundant data-storage solutions will always involve a cost v risk trade-off. The issue is not whether you backup data (that’s a given) but of aligning risk mitigation with cost.’
‘If you are designing a solution for which a third-party (eg SaaS provider) will manage your data, you must understand the company’s backup policies/procedures, and even then you will probably still want to integrate your own.’

Slide 5. existing/future capacity (designing for).
‘If moving an application to the cloud, you must monitor it closely to fully understand its processing attributes (user demand/CPU use/RAM use, data storage consumption). Knowing this will help you guess your system’s cloud resource needs’.
‘With this knowledge, you can design for scalability (the ease of integrating extra computing resources). Two types: vertical scaling – scale *up* by moving the app to a faster, more powerful processor. Horizontal scaling – scale *out* by distributing different tasks across different servers.’

Slide 6. Configuration management (designing for).
‘you cloud-based app can be used any time and by any device/browser. So you must consider a variety of Oss, browsers, device-specific GUIs etc. Oss and browsers often need patches to address security issues, and each will eventually face new version release. When designing, you will want to layer configuration settings on top of your system [TOP LAYER/APPLICATION LAYER], This way, you will reduce the impact to a computer-based user when changes are made to a handheld device etc
‘If you use an SaaS provider, you need to know the company’s patch management and versioning policies/procedures’.

Slide 7. Deployment (designing for).
‘Desktop virtualisation is changing how solutions are delivered. From an OS on demand to thin client (browser-based) solutions, developers have myriad of ways to deploy a system. As you design a system, you should identify each potential user type and its environment attributes (eg OS, device type, browser). Then you need to consider not only how you will deploy the initial solution to the user, but also how you will deploy system upgrades.’

Slide 8. Disaster recovery (designing for).
‘When designing a solution with respect to disaster recovery/business continuity, you must balance risks with costs. It is impossible /unecessary to protect a system from all potential events. Instead, you must determine each event’s likelihood and business impact, and then seek to provide an affordable solution that mitigates risks.
Fortunately, the cloud’s affordable and distributable resources provide developers with considerable flexibility’

Slide 9. Environment (designing for).
In a data centre, the biggest environmental impact is power consumption to drive devices/air conditioner. As more companies migrate to PaaS/IaaS providers, many smaller, and possibly less efficient data centres are being accumulated into larger, state-of-the-art facilities.

Slide 9. Interoperability (designing for).
‘Today, many companies use a wide-range of cloud solutions (eg for CRM, HR, etc). To simplify user interactions with such solutions, many companies strive to integrate the solutions and often even share data across the solutions
‘In the past, companies would buy and install middleware to facilitate the exchange of data between solutions.
Today there are cloud-based middleware solutions that let companies tie together two cloud-based solutions, often without the need for programming development.
‘As you design, consider ways you may need to integrate data between applications and then design accordingly.’

Slide 10. Maintainability (designing for).
‘The most costly phase of software development cycle is the system maintenance phase.,
‘To maximise code reuse and increase code maintainability, software engineers are taught to create highly function (cohesive) and independent (loosely coupled) software modules.
‘If you are using an SaaS, you need to keep the long-term nature of your relationship in mind. Many people argue that cloud solutions are initially cheap but may cost you more in the long run’

Slide 11. Performance (designing for).
‘Designing for performance means optimising what you have (as opposed to designing for scalability, which is designing for future use of extra resources).

- Reduce the use of graphics on key pages ...
- Optimise the graphics file format for all images
- Compress large text blocks before downloading
- Utilise data and application caching
- Fine-tune disk and database I/O operations
- Reduce network operations
- Fine-tune secure data communication transactions

Slide 12. privacy (designing for).
'You must protect a user's data privacy. Especially if you are developing healthcare/e-commerce solution that stores credit card info
'You will need to design your solution in a way that protects data not only from external access, but also from internal users eg developers/administrators
'Backup/replicating databases is v important, but each data backup creates a potential opportunity for a user/admin/hacker to gain access to the data.

Slide 13. portability (designing for).

'Portability = the ease with which a solution can be moved, typically from one platform to another.
Ideally you should design a solution so it can easily move from one cloud provider to another.
Many argue that using open-source tools can increase an application's portability
If you are designing your own solution, be aware that provider-specific APIs, which may not be available through other providers, may create vendor lock-in'

Slide 14. Recovery/Reliability/Robustness (designing for).

In addition to disaster recovery, you should also design your system to recover from more common events: server failure, user error, power outage.

Computing devices will eventually fail. They have MTBF (mean time before failure). You can use this attribute to estimate the device's life expectancy. As you design your solutions, you must identify potential signal points of failure and then design potential system redundancy or establish an acceptable system downtime.

Robustness = measure of a site's ability to continue in the event of error or system failure.
As you design, you should strive to identify and eliminate single points of failure.
You should also consider *automating a system a resource utilisation monitor* that *alerts administrators* before a system's resources become critically low

Slide 15. Response time (designing for).
Keep user experience in mind when designing (they will leave if it takes more than a few seconds to download).
As you design you need to consider:
- page download times
- the system response time after a user operation (eg submitting a form)
Your response time design efforts may be related to your site's capacity plan design.
Companies exist that test user experience – they will evaluate a system from different locations, using different connection speeds and browsers.

Slide 16. Security (designing for).
Developers must consider:
- Software patch installations and software version management

- Early awareness of security incidents
- Data privacy
- Jurisdictional issues for a remote cloud provider... (eg GDPR)
- Multi-tenant solution issues ...sharing servers
- Cloud-provider failure or collapse
- Defense mechanisms for common low-level network attacks
- Data wiping for share-storage space
- Physical security considerations

Slide 16. Testability and usability (designing for).
‘Cloud based solutions have functional/non-functional requirements. As you design, you need to keep in mind how you will test both aspects. Non-functional requirements are often the hardest to test. Some developers use Test-driven design...’

Usability = a measure of the system’s ease of use. System must be usable.
Because of the importance of meeting system usability requirements, many designers will model or create a prototpye of the user experience so they can receive user feedback early in the design process.’