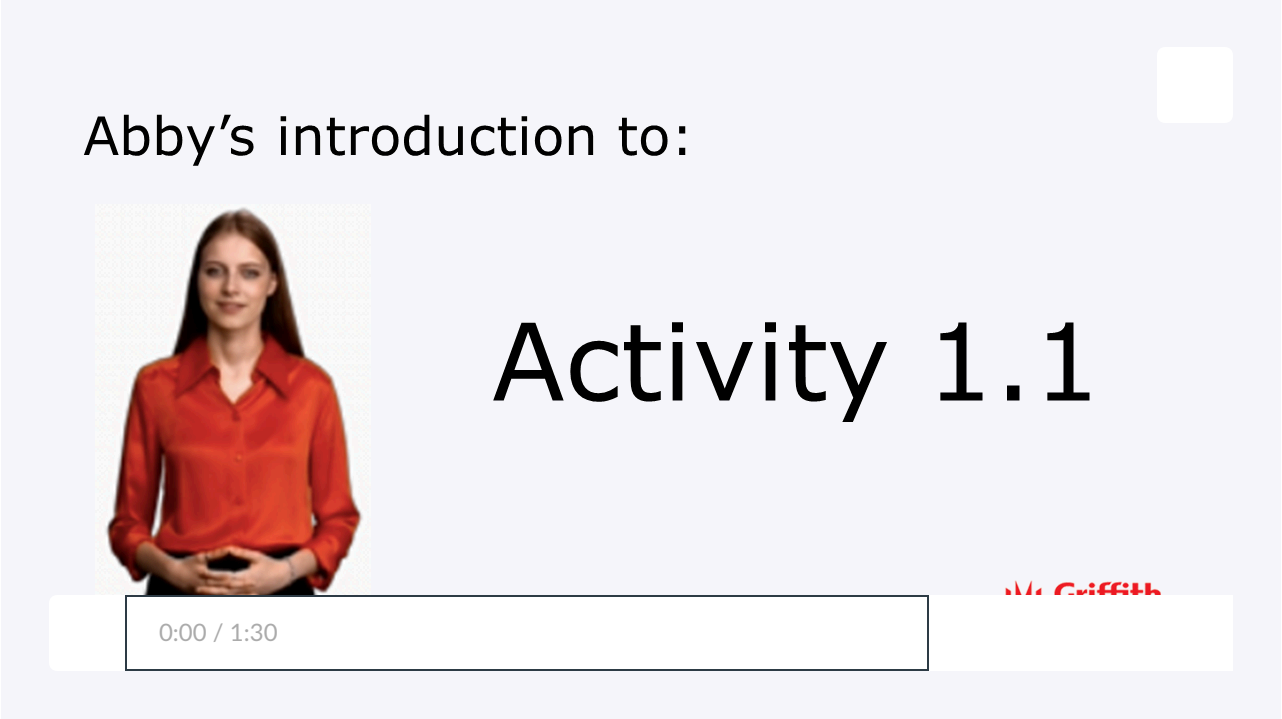
Activity 1.1 Analyse requirements-gathering scenarios and propose appropriate techniques

Access course FAQ chatbot (https://lms.griffith.edu.au/courses/24045/pages/welcome-to-the-course-chatbot)

Module 1 - Gather and analyse requirements



What is this activity?

In Activity 1.1, you will be introduced to requirements-gathering for designing an application system. Gathering requirements from stakeholders is the first step in understanding what it is that an application system should do (i.e., its business function). Firstly, we will go through a case study where you need to identify the key stakeholders, objectives, and potential requirements for an application system. The content will also introduce you to different types of requirements elicitation techniques and how to evaluate their suitability based on the types of stakeholders and the information you're attempting to

gather. You will finish by exploring the assignment scenarios and selecting one that best suits your interests.

Why is this activity important?

This activity will develop your ability to navigate the complexities of requirements-gathering in real-world application system design projects. By analysing the case study, you will gain practical experience in applying requirements-gathering concepts and best practices. You will develop your critical thinking and decision-making skills in the context of application system design and learn to adapt your approach to requirements-gathering based on project-specific factors and constraints.

Case study

▼ EduLink - Learning Management System

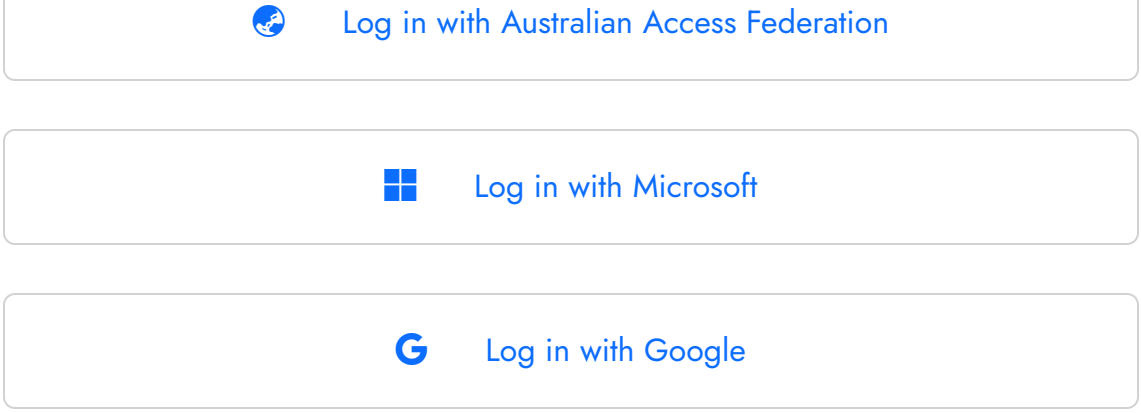
Scenario:

EduLink, a global education provider, is planning to develop a Learning Management System (LMS) that will support online course delivery, student collaboration, and assessment management. The LMS will need to integrate with existing student information systems, support personalised learning experiences, and provide educators with tools to create engaging course content. You have been hired as the lead application system designer for EduLink's LMS project.

Your job:

- Identify the stakeholders for the project - Supporting content A and C
- Outline the requirements for each identified stakeholder group - Supporting content A, C and E
- Propose the most suitable requirements elicitation technique for each stakeholder group - Supporting content C, D and E
- Consider the challenges and constraints specific to the education domain, such as accessibility requirements and the need to support diverse learning styles - Supporting content B

Use the chatbox below to begin engaging with the exercise.



If your browser fails to open the agent above, you can access it directly via **this link (https://app.cogniti.ai/agents/66471009f2a1d77ddd7c87b8/chat?k=V_R_qLXhH_wpjEU-8G6jVkh6PQN9hRzzh23sqt7cC_M)** .

A note on using AI tutors:

This AI tutor is designed to support your learning journey, providing a flexible opportunity you can engage with anytime, anywhere. It's not marked – the goal is purely to enhance your understanding and skills at your own pace. To gain the most from this experience, keep these three principles in mind:

1. Embrace the learning process: Treat mistakes as opportunities to grow, not failures to avoid.
2. Engage honestly: Approach the AI as a helpful mentor, not a system to outsmart.
3. Apply and reflect: Use the AI's feedback to improve your work and deepen your understanding.

Remember, this is your space to practise, explore, and learn without pressure. It's up to you to make the most of it.

Supporting content for this activity

Use the supporting content below to assist you as you engage with the case study activity. These will provide you with the knowledge and tools that you need to successfully complete this activity.

▼ Supporting content A - Fundamentals of requirements-gathering

Key concepts and terminology

In the context of requirements-gathering for application systems, there are several key concepts and terminology that are fundamental to understanding and executing the process effectively. Here are some of the most important terms and concepts:

1. **Requirements Gathering**: The process of collecting the requirements or conditions that a particular system or software application must satisfy. This involves working with stakeholders to understand their needs and documenting these needs in a clear and concise manner.
2. **Stakeholders**: Individuals or groups who have an interest or stake in the system being developed. Stakeholders can include end-users, customers, developers, project managers, and other interested parties, e.g., sponsors or governing bodies.
3. **Functional Requirements**: Specific behaviors or functions that the system must perform. These are typically described in terms of inputs, processing, and outputs.
4. **Non-Functional Requirements (NFRs)**: Constraints or qualities that the system must have, such as performance, security, reliability, usability, and maintainability. These are often referred to as the "ilities."
5. **User Stories**: A technique used in Agile software development to describe a requirement from the perspective of the end-user. A user story typically follows a simple template: "As a [type of user], I want [some feature] so that [some benefit]."
6. **Use Cases**: A description of a system's behaviour as it responds to a request that comes from outside of that system. Use cases define the interactions between external actors and the system.

7. **Prototyping**: The creation of a preliminary model of the system or parts of the system to visualise and test certain aspects of the design. This can help in gathering feedback from stakeholders and refining requirements.
8. **Requirements Elicitation**: The process of discovering and documenting the requirements for a system. This involves interviewing stakeholders, observing work processes, and using various techniques to uncover the needs and expectations of the users.
9. **Requirements Analysis**: The process of examining and refining the gathered requirements to ensure they are complete, consistent, and feasible. This step often involves modeling the requirements to better understand their implications.
10. **Requirements Specification**: The formal documentation of the system's requirements. This document serves as a contract between the stakeholders and the development team, outlining what the system will do.
11. **Requirements Validation**: The process of ensuring that the requirements reflect the stakeholder's actual needs and expectations. This involves reviewing and testing the requirements to confirm their correctness.
12. **Requirements Management**: The ongoing process of managing changes to the requirements throughout the project lifecycle. This includes tracking, documenting, and prioritising changes to ensure that the system continues to meet stakeholder needs.
13. **MoSCoW Method**: A requirements prioritisation technique that stands for Must have, Should have, Could have, and Won't have this time. It helps in categorising requirements based on priority and necessity.
14. **Traceability**: The ability to link product requirements to their origin and to track their evolution over time. Traceability ensures that each requirement can be traced back to its source and that changes are documented and understood.

Understanding these concepts and terminology is important for anyone involved in the requirements-gathering process, as they form the foundation of effective communication and collaboration between stakeholders and the development team.

The importance of requirements gathering in application system design

Requirements gathering is a critical phase in the design of application systems as it lays the foundation for the entire development process. It involves the systematic collection of information about the needs of the end-users, the system's functionality, and the constraints under which it must operate. This phase is essential because it ensures that the final product meets the **expectations of the stakeholders** and fulfills its **intended purpose**. Without thorough requirements gathering, it is

likely that the system will not address the needs of its users, leading to dissatisfaction, increased costs due to rework, and potential project failure.

Stakeholders (**Image source** (https://teamboard.cloud/stakeholders-in-project-management/))

The importance of requirements gathering cannot be overstated as it directly impacts the **quality and success of the application system**. It helps in identifying and understanding the **problems** that the system is supposed to solve, the **environment** in which it will operate, and the **specific features** that are necessary for its effectiveness. By engaging in comprehensive requirements gathering, project teams can avoid scope creep, where the project's requirements grow beyond the original plan, often leading to delays and budget overruns. Moreover, it allows for the early detection of potential issues and conflicts among requirements, enabling the team to address them before they escalate into larger problems.

Furthermore, requirements gathering facilitates **better communication and collaboration among stakeholders**, including users, developers, and project managers. It provides a common understanding of the project's goals and objectives, ensuring that everyone is aligned and working towards the same vision. This shared understanding is crucial for making informed decisions throughout the development process and for managing expectations. When stakeholders are involved in the requirements-gathering process, they feel more invested in the project, which can lead to greater acceptance of the final product.

In addition to these benefits, requirements gathering also supports the **creation of a detailed requirements document**, which serves as a roadmap for the development team. This document outlines the system's functionality, user interactions, data requirements, and performance criteria, among other aspects. It acts as a reference throughout the development lifecycle, guiding the design, implementation, testing, and deployment of the application system. A well-documented set of requirements also aids in the estimation of project timelines and costs, making it easier to manage resources effectively and deliver the system on time and within budget.

The requirements-gathering process

The requirements-gathering process is a systematic approach to understanding and documenting the needs and expectations of stakeholders for an application system. This process is iterative and involves several key steps that help ensure the system's design will meet the intended goals and objectives. The first step typically involves **identifying and engaging with stakeholders**, which includes end-users, customers, developers, and other interested parties. Through interviews, surveys, workshops, and observation, the requirements-gathering team seeks to elicit the needs and desires of these stakeholders.

Requirements gathering (Image source (https://www.geeksforgeeks.org/requirements-gathering-introduction-processes-benefits-and-tools/))

Once stakeholders have been identified, the next step is to gather **detailed information about the requirements**. This can involve the use of various techniques such as brainstorming sessions, storyboarding, and the creation of user stories or use cases. These techniques help in understanding the context in which the system will be used, the tasks it needs to perform, and the environment it will operate within. The information collected during this phase is crucial as it forms the basis of the system's functionality and features.

Following the initial collection of requirements, the next step is to **analyse and refine** them. This involves reviewing the gathered information to ensure clarity, completeness, and feasibility. Requirements are often **prioritised** using methods such as the MoSCoW technique (Must have, Should have, Could have, Won't have) to determine which features are essential and which can be considered for future releases. This phase also involves identifying any conflicts or dependencies between requirements and resolving them through negotiation and compromise.

The final step in the requirements-gathering process is to **document the requirements** in a clear and structured manner. This documentation serves as a reference for the development team and stakeholders throughout the project lifecycle. It should be detailed enough to guide the design and implementation of the system but also flexible enough to accommodate changes as the project evolves. Effective requirements documentation includes both functional requirements, which describe what the system should do, and non-functional requirements, which specify the system's

performance, security, and usability characteristics. Throughout the development process, the requirements document is revisited and updated to reflect any changes or new insights gained, ensuring that the system remains aligned with stakeholder needs.

▼ Supporting content B - Examples of domain-specific considerations

Healthcare: Privacy regulations, patient safety, and clinical workflows

In the healthcare domain, requirements gathering must be approached with a deep understanding of the unique challenges and regulations that govern patient privacy, safety, and clinical workflows. Here's an example of how requirements gathering could be conducted in this context:

1. **Stakeholder Interviews and Workshops:**
    - **Clinical Staff:** Nurses, doctors, and other healthcare professionals are interviewed to understand their daily workflows, the information they need to access, and how they interact with existing systems.
    - **IT Staff:** IT personnel are consulted to understand the technical infrastructure, integration points, and any technical constraints.
    - **Patients and Caregivers:** Feedback is gathered to understand their needs for accessing health information, scheduling appointments, and interacting with the healthcare system.
    - **Regulatory Compliance Officers:** These stakeholders ensure that the system complies with privacy regulations such as HIPAA (Health Insurance Portability and Accountability Act) in the United States, GDPR (General Data Protection Regulation) in the European Union, or AHPRA (Australian Health Practitioner Regulation Agency) in Australia.
2. **Document Analysis:**
    - Review existing documentation such as patient records, clinical protocols, and privacy policies to identify requirements related to data handling, access controls, and reporting.
3. **Observation of Workflows:**
    - Conduct on-site observations of clinical workflows to see firsthand how healthcare professionals interact with patients and technology. This can reveal hidden requirements and pain points that might not be evident in interviews alone.
4. **Use Cases and Scenarios:**

- Develop detailed use cases and scenarios that describe how different users will interact with the system. For example, a use case might describe how a doctor views a patient's medical history or how a patient schedules an appointment online.

5. **Requirements Workshops:**
   - Organise collaborative workshops with representatives from all stakeholder groups to refine requirements, resolve conflicts, and ensure that the system design meets the needs of all users while adhering to privacy regulations and safety standards.
6. **Prototyping:**
   - Create prototypes of the system to allow stakeholders to interact with potential solutions and provide feedback early in the development process. This can help in validating requirements and identifying any necessary adjustments.
7. **Regulatory Review:**
   - Involve legal and compliance experts to review the requirements to ensure they meet all relevant healthcare regulations, including those related to patient privacy and data security.
8. **Risk Assessment:**
   - Conduct a risk assessment to identify potential risks to patient safety and privacy that could arise from the system. This includes assessing the impact of system failures or security breaches.
9. **Traceability Matrix:**
   - Create a traceability matrix to link requirements to specific stakeholder needs, regulatory standards, and system functions. This helps in ensuring that all requirements are justified and necessary.
10. **Iterative Validation:**
    - Continuously validate requirements with stakeholders throughout the development process to ensure they remain relevant and aligned with the evolving needs of the healthcare organisation and regulatory landscape.

By following these steps, the requirements-gathering process in the healthcare domain can be comprehensive, ensuring that the resulting application system is not only functional but also compliant with privacy regulations, safe for patients, and supportive of efficient clinical workflows.

Finance: Security, compliance, and data integrity

In the finance domain, requirements gathering must be meticulous to ensure that the application systems address the critical aspects of security, compliance with financial regulations, and data

integrity. Here's an example of how requirements gathering could be conducted in this context:

1. **Stakeholder Analysis:**
    - Identify and engage with key stakeholders, including financial analysts, traders, compliance officers, IT security specialists, and executives.
    - Understand their roles, the data they interact with, and their specific needs and concerns regarding security, compliance, and data integrity.
2. **Regulatory Review:**
    - Research and compile a list of relevant financial regulations and standards (e.g., Sarbanes-Oxley Act, General Data Protection Regulation, Payment Card Industry Data Security Standard) that the system must comply with.
    - Involve compliance officers to ensure that all regulatory requirements are captured and understood.
3. **Security Assessment:**
    - Conduct a thorough security assessment to identify potential threats and vulnerabilities.
    - Work with IT security specialists to define security requirements, such as encryption standards, access controls, multi-factor authentication, and audit trails.
4. **Data Analysis:**
    - Analyse the types of financial data that will be handled by the system, including sensitive information like account details, transaction records, and personal data.
    - Determine the data integrity requirements, such as data validation rules, backup procedures, and recovery time objectives in case of data loss.
5. **Business Process Mapping:**
    - Map out the current financial business processes to understand how data flows through the organisation.
    - Identify any inefficiencies or risks related to data handling and security.
6. **Use Case Development:**
    - Develop detailed use cases that describe how users will interact with the system, including scenarios for data entry, reporting, transaction processing, and compliance checks.
7. **Workshops and Interviews:**

- Organise workshops and conduct interviews with stakeholders to discuss the findings from the analysis and to gather additional requirements.
- Use techniques like storyboarding and prototyping to facilitate discussions and gather feedback.

8. **Requirements Prioritisation:**
   - Prioritise the requirements based on their impact on security, compliance, and data integrity, as well as their alignment with business goals.
9. **Documentation:**
   - Document all requirements in a clear and structured format, including acceptance criteria and traceability to specific stakeholder needs and regulatory standards.
10. **Review and Validation:**
    - Review the requirements with stakeholders to validate their accuracy and completeness.
    - Update the requirements as necessary based on feedback.
11. **Change Management:**
    - Establish a change management process to handle new or changing requirements throughout the system development lifecycle.

By following these steps, the requirements-gathering process in the finance domain can ensure that the application system is designed with a strong focus on security, compliance, and data integrity, which are critical for the financial industry.

E-commerce: User experience, scalability, and payment processing

In the e-commerce domain, requirements gathering must focus on delivering a seamless user experience, ensuring the system can scale to handle varying loads, and providing secure and reliable payment processing. Here's an example of how requirements gathering could be conducted in this context:

1. **Stakeholder Engagement:**
   - Identify and engage with key stakeholders, including business owners, marketing specialists, customer service representatives, IT staff, and payment processing partners.

- Understand their perspectives on user experience, scalability needs, and payment processing requirements.

2. **User Research:**
   - Conduct user research through surveys, interviews, and usability testing to gather insights into the target audience's preferences, pain points, and expectations regarding the shopping experience.
3. **Competitive Analysis:**
   - Analyse competitor e-commerce platforms to identify industry standards and best practices for user experience, scalability, and payment processing.
4. **User Experience (UX) Design Workshops:**
   - Organise workshops with UX designers, developers, and stakeholders to brainstorm and prototype potential user interfaces and experiences.
   - Use wireframing and mockups to visualise the user journey and gather feedback.
5. **Functional Requirements:**
   - Define functional requirements for the e-commerce platform, such as product catalog management, search functionality, shopping cart behaviour, checkout process, and post-purchase communication.
6. **Scalability Planning:**
   - Assess the expected user traffic and transaction volumes to determine the scalability requirements.
   - Work with IT and DevOps teams to define technical specifications for a scalable architecture that can handle peak loads, such as during sales events.
7. **Payment Processing Requirements:**
   - Collaborate with payment processing partners to understand the integration requirements, security protocols (e.g., PCI DSS compliance), and supported payment methods.
   - Define requirements for handling payment transactions, including encryption, tokenization, and fraud detection mechanisms.
8. **Performance Metrics:**
   - Establish key performance indicators (KPIs) and benchmarks for system performance, such as page load times, transaction speed, and uptime.
9. **Legal and Compliance Review:**

- Review legal and compliance requirements related to e-commerce, such as data protection laws (e.g., GDPR), consumer rights, and tax regulations.

10. **Documentation and Prioritisation:**
    - Document all gathered requirements in a structured format, including user stories, acceptance criteria, and prioritisation based on business goals and customer needs.
11. **Review and Validation:**
    - Conduct review sessions with stakeholders to validate the requirements and ensure they align with the overall business strategy and customer expectations.
12. **Prototyping and Usability Testing:**
    - Develop interactive prototypes to demonstrate the user experience and gather feedback on the proposed design and functionality.

By following these steps, the requirements-gathering process in the e-commerce domain can ensure that the application system is designed with a focus on delivering an exceptional user experience, is capable of scaling to meet demand, and offers secure and efficient payment processing capabilities.

Education: Accessibility, personalised learning, and assessment management

In the education domain, requirements gathering must focus on ensuring that technology is accessible to all learners, supports personalised learning experiences, and effectively manages assessments. Here's an example of how requirements gathering could be conducted in this context:

1. **Stakeholder Interviews and Workshops:**
    - Engage with educators, students, administrators, and IT staff to understand their needs and challenges related to accessibility, personalised learning, and assessment management.
    - Use workshops to collaboratively explore potential solutions and gather detailed requirements.
2. **Accessibility Audit:**
    - Conduct an audit of existing educational materials and platforms to identify accessibility barriers for students with disabilities.

- Gather requirements for making content and interfaces compliant with accessibility standards (e.g., WCAG - Web Content Accessibility Guidelines).

3. **Personalised Learning Requirements:**
   - Interview students to understand their learning preferences and how they would like to receive personalised educational content.
   - Define requirements for adaptive learning technologies that can adjust to individual student needs, learning styles, and performance.
4. **Assessment Management:**
   - Work with educators to understand the types of assessments they use (e.g., quizzes, essays, projects) and how they track student progress.
   - Gather requirements for tools that can automate assessment creation, delivery, grading, and feedback provision.
5. **Legal and Compliance Review:**
   - Review legal requirements related to education technology, such as data protection laws (e.g., FERPA in the United States, Australian Privacy Act 1988) and accessibility regulations.
   - Ensure that all requirements align with these legal frameworks.
6. **Technical Feasibility Assessment:**
   - Consult with technical experts to assess the feasibility of the gathered requirements.
   - Determine the technical specifications needed to support accessibility features, personalised learning, and robust assessment management.
7. **User Experience (UX) Design Input:**
   - Collaborate with UX designers to create user personas and journey maps that reflect the diverse needs of users in the education system.
   - Use design thinking workshops to generate ideas for intuitive interfaces that support accessibility and personalised learning.
8. **Integration Requirements:**
   - Identify the need for integration with other educational systems (e.g., student information systems, learning management systems) to ensure a seamless experience.
   - Define the technical requirements for these integrations.
9. **Data Privacy and Security:**
   - Gather requirements for data handling, ensuring that student data is collected, stored, and used in compliance with data protection regulations.
   - Define security measures to protect sensitive student information.

10. **Documentation and Prioritisation:**
    - Document all requirements in a clear format, including user stories, use cases, and acceptance criteria.
    - Prioritise requirements based on their impact on accessibility, personalised learning, and assessment management, as well as their alignment with educational goals.
11. **Prototyping and User Testing:**
    - Develop prototypes of the proposed educational tools and conduct user testing with students and educators.
    - Gather feedback to refine requirements and ensure they meet user needs.
12. **Review and Validation:**
    - Present the gathered requirements to stakeholders for validation and approval.
    - Make adjustments based on feedback to ensure the final requirements specification is comprehensive and actionable.

By following these steps, the requirements-gathering process in the education domain can ensure that the application system supports accessibility, personalised learning, and effective assessment management, ultimately enhancing the educational experience for all users.

▼ Supporting content C - Elicitation techniques

Interviews: Conducting effective stakeholder interviews

Interviews (Image source (https://www.softwaretestinghelp.com/requirements-elicitation-techniques/))

Conducting effective stakeholder interviews is an important component of the requirements-gathering process in application systems development. Stakeholders possess valuable insights into the business processes, user needs, and system requirements that are essential for designing a

successful application. To ensure that these interviews are effective, it is important to prepare thoroughly. This preparation includes identifying the right stakeholders, understanding their roles and responsibilities within the organisation, and determining the key questions that need to be asked to elicit the necessary information. By tailoring the interview approach to the stakeholder's expertise and perspective, interviewers can create an environment that encourages open and informative dialogue.

During the interview, establish a rapport with the stakeholder to facilitate a comfortable and **collaborative** discussion. **Active listening** skills are vital; interviewers should demonstrate genuine interest in the stakeholder's responses, ask clarifying questions, and avoid interrupting. It is also important to be mindful of the stakeholder's **time constraints** and to ensure that the interview stays on track while allowing for the exploration of important topics that may emerge unexpectedly. Effective note-taking or, with permission, audio recording of the interview can help capture the essence of the conversation, ensuring that no critical details are missed.

Post-interview, the information gathered needs to be analysed and synthesised into **actionable insights**. This involves reviewing the notes or recordings, identifying key requirements, and reconciling any discrepancies in the information provided by different stakeholders. It is often beneficial to share a summary of the interview findings with the stakeholders for validation and to ensure that their input has been accurately interpreted. This feedback loop not only confirms the accuracy of the gathered requirements but also strengthens the relationship with the stakeholders, fostering trust and collaboration throughout the application development lifecycle.

Focus Groups: Facilitating productive focus group discussions

Focus groups (**Image source** (https://www.questionpro.com/blog/focus-group/))

Facilitating productive focus group discussions is an art that requires careful planning and skilled execution. Focus groups bring together a diverse set of stakeholders and potential users to discuss their needs, expectations, and perceptions regarding an application system. The success of these discussions largely depends on the **preparation** beforehand. This includes defining clear objectives

for the focus group, selecting participants who can contribute meaningfully to the discussion, and creating a discussion guide that outlines key topics and questions. The guide should be structured to encourage open dialogue while ensuring that all necessary areas are covered. It is also important to choose a neutral and comfortable venue that promotes an open and relaxed atmosphere.

During the focus group session, the facilitator plays a pivotal role in guiding the conversation, **keeping it on track**, and ensuring that all participants have the opportunity to contribute. The facilitator must be an active listener, attentive to both the content of what is being said and the dynamics of the group. They should encourage participants to elaborate on their comments, ask probing questions to gain deeper insights, and manage any digressions tactfully. Moreover, the facilitator should be adept at handling sensitive topics or strong opinions in a way that maintains a respectful and constructive environment. Effective **note-taking** or, when appropriate, **audio recording** is essential to capture the richness of the discussion without relying solely on memory.

After the focus group, the data collected must be analysed to extract **meaningful insights**. This involves reviewing the notes or recordings, identifying common themes, and assessing the implications for the application system. The facilitator or a team of analysts should look for **patterns** in the feedback, noting both the consensus opinions and dissenting views. This analysis should be synthesised into a **report** that not only summarises the findings but also provides recommendations for how the feedback can be incorporated into the system design. Sharing these findings with the participants and other stakeholders can validate the results, provide additional insights, and foster a sense of involvement and ownership in the development process. It is through this comprehensive approach that focus group discussions can become a powerful tool in eliciting requirements and informing the design of application systems.

Surveys: Designing and distributing surveys to gather user requirements

Surveys (**Image source** (https://www.e-ir.info/2021/08/25/surveys/))

Designing and distributing surveys to gather user requirements is a strategic approach to eliciting a wide range of feedback from a large number of stakeholders efficiently. The effectiveness of this method hinges on the **meticulous design** of the survey instrument. It is crucial to craft questions that are clear, concise, and unbiased, avoiding jargon and complex language that could confuse respondents or lead to inaccurate interpretations. The survey should be **structured** in a logical flow, starting with general questions to ease participants into the topic before delving into more specific areas. Including a mix of question types, such as multiple-choice, rating scales, and open-ended questions, can provide both quantitative data for analysis and qualitative insights into user needs and preferences.

When distributing surveys, it is important to consider the **target audience** and the most appropriate channels for reaching them effectively. Email lists, social media platforms, and dedicated survey websites are common methods for dissemination. The invitation to participate should clearly communicate the purpose of the survey, the estimated time required to complete it, and any **incentives** for participation, such as entry into a prize draw or a summary of the findings. It is also beneficial to ensure that the survey is **accessible on various devices**, as respondents may use desktops, laptops, tablets, or smartphones to complete it. Following up with **reminders** to non-respondents can increase the response rate and the representativeness of the data collected.

After the survey closes, the **analysis** of the data is a critical step in transforming raw responses into actionable insights. Quantitative data can be analysed using statistical methods to identify patterns, trends, and correlations. Qualitative data from open-ended questions requires a more interpretive approach, often involving coding and thematic analysis to discern common themes and sentiments. It is important to consider the response rate and the **representativeness of the sample** when interpreting the results to ensure that the findings are not skewed by a low or non-representative response. Once analysed, the results should be documented in a comprehensive **report** that not only presents the data but also provides recommendations for how the insights can be integrated into the application system design. Disseminating the findings back to the survey participants and other stakeholders can help validate the results and foster engagement in the development process.

Workshops: Collaborating with stakeholders to elicit and prioritise requirements

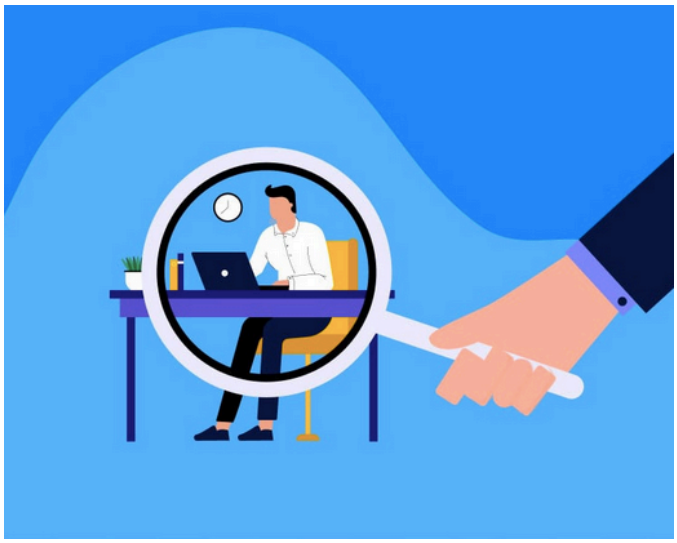
Workshops (Image source (https://thebadoc.com/ba-techniques/f/requirements-workshop-that-works))

Collaborating with stakeholders to elicit and prioritise requirements is a fundamental aspect of the requirements-gathering process in application systems development. Effective collaboration ensures that the system being developed meets the needs of all stakeholders, from end-users to business executives. The first step in this collaboration is to **identify and engage** all relevant stakeholders, understanding their roles, interests, and expectations. This can be achieved through initial meetings or workshops where stakeholders are introduced to the project and its objectives, and where the importance of their input is emphasised.

Eliciting requirements from stakeholders is an **iterative process** that involves various techniques such as interviews, workshops, questionnaires, and prototyping. During these activities, it is crucial to create an environment where **stakeholders feel heard** and valued, encouraging them to share their insights, concerns, and ideas openly. The information gathered should be documented clearly and shared with stakeholders for validation, ensuring that their requirements are accurately captured. As new requirements emerge and are added to the list, the need for prioritisation becomes evident.

Prioritising requirements is a collaborative effort that involves working with stakeholders to understand the relative importance and urgency of **each requirement**. This can be done through techniques such as **MoSCoW analysis** (Must have, Should have, Could have, Won't have this time), **analytical hierarchy** process (AHP), or by assigning **priority scores** based on criteria such as business value, user impact, and feasibility. It is important to facilitate discussions among stakeholders to resolve conflicts in priorities and to ensure that the final prioritised list reflects a consensus that aligns with the project's goals and constraints. The outcome of this collaboration is a **prioritised set of requirements** that guides the development process, ensuring that the most critical needs are addressed first while keeping the project on track and within budget.

Ethnographic Observation: Observing users in their natural environment



Ethnographic observation (Image source (https://www.netsolutions.com/insights/how-to-do-ethnographic-research/))

Ethnographic observation is a qualitative research method that involves observing users in their natural environment to gain insights into their behaviours, interactions, and needs related to application systems. This technique is particularly useful for understanding the context in which users will be interacting with the system and for identifying requirements that may not be evident through other elicitation methods. The key to successful **ethnographic observation** is to immerse oneself in the user's environment without interfering with their natural behaviour, thus capturing authentic and spontaneous interactions.

When conducting ethnographic observation, researchers must be prepared to spend a significant amount of **time** with users to ensure they capture a comprehensive range of activities and scenarios. This often requires establishing a rapport with the users to make them comfortable with the presence of an observer. Researchers should be **unobtrusive**, taking notes or using audio and video recording devices to document observations without disrupting the flow of activities. It is also important to be **sensitive** to the users' privacy and to obtain informed consent before recording any data.

The data collected through ethnographic observation is rich and can provide a deep understanding of user experiences and requirements. **Analysis** of this data involves identifying patterns, themes, and critical incidents that reveal user needs, pain points, and potential opportunities for system improvement. Researchers may use techniques such as **coding** and **thematic analysis** to organise and interpret the observations. The **findings** from ethnographic observation can be particularly valuable for informing the design of user interfaces, workflows, and system features that are aligned with how users naturally perform their tasks. This method helps ensure that the application system is not only functional but also intuitive and user-friendly, enhancing user satisfaction and productivity.

▼ Supporting content D - Evaluating the suitability of elicitation techniques

Factors to consider when selecting elicitation techniques

When selecting elicitation techniques for gathering requirements in the development of application systems, several key factors must be considered to ensure that the techniques chosen are appropriate and effective. One crucial factor is the **complexity** and **nature** of the requirements themselves. **Different elicitation techniques** are better suited to different types of requirements; for example, while user interviews may be highly effective for understanding user needs and preferences, document analysis might be more suitable for extracting requirements from existing system artifacts. Additionally, the level of detail required for the requirements will influence the choice of technique, with more intricate

requirements potentially necessitating more interactive and iterative methods such as prototyping or joint application development (JAD).

Another significant factor is the **stakeholder involvement** and their characteristics. The availability, willingness, and ability of stakeholders to participate in the elicitation process will greatly affect the choice of techniques. Techniques that require active stakeholder engagement, such as workshops or focus groups, may not be feasible if stakeholders are not readily available or are geographically dispersed. Furthermore, the **diversity** of stakeholders, including their technical expertise and familiarity with the system, will influence the selection of elicitation methods. For instance, personas and storyboarding might be more appropriate for non-technical stakeholders to help them visualise and articulate their requirements.

The **project constraints**, including time, budget, and resources, are also critical factors in selecting elicitation techniques. Some techniques, like ethnographic studies or extensive surveys, can be resource-intensive and time-consuming, which may not be viable for projects with tight deadlines or limited budgets. In contrast, techniques like questionnaires or document reviews can be more cost-effective and efficient, although they may not provide the same depth of information. It is essential to **balance** the need for comprehensive requirements with the practical realities of the project constraints to select elicitation techniques that are both informative and feasible.

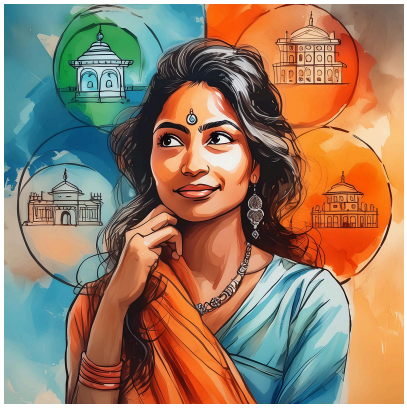
Mapping techniques to project characteristics and constraints

Mapping elicitation techniques to project characteristics and constraints is a critical step in the requirements gathering process for application systems. The goal is to **align** the chosen techniques with the unique aspects and limitations of the project to ensure that the requirements are gathered effectively and efficiently. One of the primary project characteristics to consider is the **size and complexity** of the system being developed. Large and complex systems may require a combination of elicitation techniques, such as workshops for high-level requirements and use case modeling for detailed functional requirements. In contrast, smaller projects might be adequately served by less formal techniques like interviews or group sessions.

Another key project characteristic is the **composition of the stakeholder group**. The diversity in stakeholder expertise, interests, and availability will influence the selection of elicitation techniques. For instance, if stakeholders are dispersed geographically, techniques that can be conducted remotely, such as online surveys or virtual focus groups, may be more appropriate. Similarly, if stakeholders have varying levels of technical knowledge, visual techniques like storyboarding or prototyping can help bridge the communication gap and facilitate a shared understanding of the requirements.

Constraints, such as time, budget, and available resources, also play a significant role in mapping elicitation techniques to a project. Techniques that require more time and resources, like ethnographic studies or facilitated workshops, might be ideal for projects with ample time and budget but may be impractical for those with tighter constraints. In such cases, more lightweight techniques like document analysis or questionnaires could be employed to gather requirements within the given limitations. It is essential to assess the trade-offs between the depth of requirements and the resources required to elicit them, ensuring that the chosen techniques strike a balance between comprehensiveness and feasibility.

Adapting techniques to specific domains



Adapting elicitation techniques to specific domains is important for effectively gathering requirements that are relevant and appropriate to the unique characteristics and constraints of each industry. Each domain, such as healthcare, finance, e-commerce, and education, has its own set of **standards**, **regulations**, and **user expectations** that must be considered when selecting and tailoring elicitation methods.

In the **healthcare domain**, for example, elicitation techniques must be adapted to address the critical nature of patient care, data privacy, and compliance with health regulations such as HIPAA. Techniques like storyboarding can be particularly effective in this domain, as they can help stakeholders visualise patient journeys and identify requirements related to safety, usability, and accessibility. Additionally, workshops and focus groups with medical professionals and patients can provide insights into clinical workflows and user needs that are specific to healthcare.

The **finance domain**, with its emphasis on security, accuracy, and regulatory compliance, may require adaptation of elicitation techniques to focus on risk management and control. Techniques such as document analysis can be used to review existing policies and procedures, while interviews with compliance officers and financial analysts can elicit detailed requirements related to transaction processing, reporting, and audit trails. Moreover, the use of prototyping can help stakeholders validate complex financial calculations and user interfaces before full-scale development.

E-commerce platforms, with their focus on user experience, sales conversion, and inventory management, may benefit from elicitation techniques that emphasise usability and customer engagement. Techniques like user personas and journey mapping can help in understanding the needs and behaviors of different customer segments, while A/B testing of prototypes can provide data-driven insights into the most effective design and functionality. Additionally, stakeholder interviews and surveys can gather requirements related to payment processing, shipping logistics, and customer service.

In the **education domain**, elicitation techniques should be adapted to account for the diverse needs of learners, educators, and administrators. Techniques such as **contextual inquiry** can be used to observe classroom interactions and learning environments, while interviews and focus groups with teachers and students can elicit requirements for personalised learning experiences and educational content. Furthermore, the use of scenarios and role-playing can help stakeholders envision how technology can support teaching methods and learning outcomes. Adapting elicitation techniques to the specific context of each domain ensures that the gathered requirements are not only comprehensive but also sensitive to the unique challenges and opportunities presented by each industry.

▼ Supporting content E - Best practices in requirements-gathering

Establishing clear objectives and scope

Establishing clear objectives and scope is a critical initial step in the requirements-gathering process for application systems. It involves defining what the system is intended to **achieve** and the **boundaries** within which it will operate. Clear objectives provide a roadmap for the project, ensuring that all stakeholders have a shared understanding of the goals. This helps in prioritising requirements and making decisions that align with the overall purpose of the system. By setting **clear objectives**, the project team can maintain focus and allocate resources effectively, avoiding scope creep and ensuring that the system meets its intended purpose.

Scope, on the other hand, defines what is included and excluded from the system. It provides a **framework** within which the system will be developed, preventing the inclusion of unnecessary features that can complicate the project and delay its completion. A well-defined scope helps in managing stakeholder expectations by clearly outlining what the system will and will not do. It also facilitates the creation of accurate estimates for time, cost, and resources, as it provides a **clear boundary** for the work that needs to be done. Establishing a clear scope early in the requirements-gathering process is essential for the successful delivery of an application system that meets the needs of its users without exceeding budgetary or temporal constraints.

Identifying and engaging key stakeholders

Identifying and engaging key stakeholders is a crucial aspect of the requirements-gathering process for application systems. **Stakeholders** are individuals or groups who have an interest in or will be affected by the system being developed. They can include end-users, managers, IT personnel, and even external parties such as clients or regulatory bodies. Effective stakeholder engagement ensures

that the requirements gathered are comprehensive and reflect the diverse needs and expectations of all interested parties.

The first step in engaging stakeholders is to **identify** who they are. This involves conducting a stakeholder analysis to determine the individuals or groups who should be involved in the requirements-gathering process. The analysis should consider the stakeholder's interest in the project, their influence, and the impact the project will have on them. Once identified, stakeholders should be **categorised** based on their level of interest and influence to prioritise engagement efforts effectively.

Engaging stakeholders requires a proactive and inclusive approach. **Communication** is key; stakeholders should be informed about the project's objectives, the requirements-gathering process, and how their input will be used. Workshops, interviews, surveys, and focus groups are common techniques used to gather requirements from stakeholders. These methods allow stakeholders to provide input, ask questions, and express concerns. It is important to create an environment where stakeholders feel **heard and valued**, as this encourages open and honest feedback. Additionally, maintaining **regular contact** with stakeholders throughout the project helps to ensure that their needs continue to be met and that any changes in requirements are promptly addressed. Effective stakeholder engagement not only leads to better system design but also fosters a sense of ownership among stakeholders, which can be vital for the successful adoption and implementation of the application system.

Effectively communicating with stakeholders

Effective communication with stakeholders is the cornerstone of successful requirements gathering for application systems. It involves not just conveying information but also understanding the needs, expectations, and concerns of the stakeholders. The ability to communicate effectively ensures that all parties are on the same page, which is essential for gathering accurate and comprehensive requirements.

To **communicate effectively** with stakeholders, it is important to use clear and simple language, avoiding technical jargon that might not be understood by all parties. It is also crucial to **tailor** the communication method to the stakeholder's preferred mode of communication, whether it be face-to-face meetings, emails, video conferences, or other forms of communication. **Active listening** is another key component of effective communication, as it allows stakeholders to express their thoughts and ensures that their input is valued and considered.

Moreover, effective communication involves being **proactive and transparent** throughout the requirements-gathering process. This means keeping stakeholders informed about the progress of the project, any changes to the requirements, and how their feedback has been incorporated into the system design. Regular updates and check-ins can help build trust and maintain engagement, ensuring that stakeholders remain invested in the project's success. Additionally, it is important to **manage expectations** by setting realistic timelines and deliverables, and by being honest about any challenges or constraints that may arise.

In summary, effective communication with stakeholders during the requirements-gathering process is essential for understanding their needs, gathering accurate requirements, and ensuring their continued engagement and support. It requires clear and tailored communication, active listening, proactivity, transparency, and realistic expectation management. By mastering these communication skills, project teams can lay a solid foundation for the successful development and implementation of application systems.

Managing conflicting requirements

Managing conflicting requirements is an inevitable challenge in the requirements-gathering process for application systems, as stakeholders often have differing opinions, priorities, and needs. Effective management of these conflicts is crucial to ensure that the final system meets the expectations of all parties while remaining feasible and aligned with the project's objectives.

The first step in managing conflicting requirements is to **identify** and **understand** the source of the conflict. This involves engaging with stakeholders to clarify their needs, expectations, and the rationale behind their requirements. By gaining a deep understanding of each stakeholder's perspective, it is possible to identify the **underlying issues** that are causing the conflict. Once the conflicts are identified, they can be **categorised** based on their nature, such as technical feasibility, resource allocation, timeline constraints, or differing user needs.

Prioritisation is a key strategy in managing conflicting requirements. This involves working with stakeholders to determine the relative importance of each requirement and how it aligns with the project's objectives and constraints. Techniques such as MoSCoW (Must have, Should have, Could have, Won't have) or the Kano model can be used to prioritise requirements based on their value and feasibility. It is important to **involve key stakeholders** in the prioritisation process to ensure that their needs are considered and that they have a sense of ownership in the decisions made.

When conflicts cannot be resolved through prioritisation alone, **negotiation and trade-offs** become necessary. This requires a delicate balance between stakeholder needs and project constraints. It may involve finding creative solutions that satisfy the underlying needs of conflicting requirements or

compromising by implementing a solution that meets the core needs of all parties, even if it is not the ideal solution for any one party. Throughout this process, it is important to maintain open and transparent communication with stakeholders, keeping them informed of the rationale behind decisions and the impact on the project. By managing conflicting requirements with care and consideration, it is possible to reach a consensus that supports the successful development and implementation of the application system.

Documenting and validating requirements

Documenting and validating requirements are essential steps in the requirements-gathering process for application systems, ensuring that all stakeholders have a clear understanding of what the system should achieve and that these requirements are feasible and accurate.

Documentation of requirements serves as the foundation for system design and development. It involves creating a **detailed record** of all the requirements gathered from stakeholders, including **functional requirements** that describe what the system should do, and **non-functional** requirements that specify the system's performance, security, and usability characteristics. A well-documented set of requirements should be **clear, concise, and unambiguous**, leaving no room for interpretation that could lead to misunderstandings or errors in the development process. Effective documentation also includes use cases, user stories, or acceptance criteria that provide context and examples of how the system will be used, making it easier for developers to understand the requirements.

Validation is the process of ensuring that the documented requirements accurately reflect the stakeholders' needs and expectations. This involves reviewing the requirements with stakeholders to confirm their completeness, correctness, and feasibility. It is important to **validate requirements** early and throughout the development process to identify and resolve any issues before they can impact the project. Techniques for validation may include prototyping, creating mock-ups, or conducting walkthroughs and reviews with stakeholders. These activities help to uncover any misunderstandings or gaps in the requirements and provide an opportunity for stakeholders to provide feedback and confirm their requirements.

Moreover, validation should also consider the **traceability** of requirements, ensuring that each requirement can be linked back to its source, whether it be a stakeholder's input, a business rule, or a system constraint. This traceability is crucial for managing changes to requirements, as it allows the project team to understand the impact of any changes on the system and its stakeholders. By meticulously documenting and validating requirements, the project team can establish a solid foundation for the development of an application system that meets the needs of its users and aligns with the project's goals.

▼ Supporting content F - Example requirements-gathering template

The provided template serves as a starting point for your requirements-gathering process. Use it as a guide to structure your analysis and documentation, but feel free to adapt and customise the template based on the specific needs of your project and the unique characteristics of your chosen domain. Consider adding, removing, or modifying sections as necessary to ensure that the template effectively supports your requirements-gathering efforts.

Project Overview

Project name and description
Goals and objectives
Scope and limitations

Stakeholder Analysis

Stakeholder identification
Stakeholder roles and responsibilities
Stakeholder communication plan

Requirements Elicitation

Elicitation techniques used
Stakeholder engagement schedule
Requirements documentation (e.g., user stories, use cases)

Requirements Analysis

Prioritisation of requirements
Dependency mapping
Conflict resolution

Requirements Validation

Review and feedback process
Sign-off and approval

Next Steps

Action items and responsibilities
Timeline and milestones
Risks and mitigation strategies

This activity is complete when you have

- Engaged with the AI tutor in the EduLink case study and participated in class discussion to share your experiences and learn from others.
- Documented your analysis and recommendations for the EduLink case study in a short report (1-2 pages, or a copy of the chat transcript), which will form part of your **portfolio (https://lms.griffith.edu.au/courses/24045/pages/building-a-portfolio-for-assignment-2)** .
- Found two other students to form your assignment 2 team. Ensure you are all enrolled in the **same course code** (i.e., 1803ICT or 7610ICT).
- Selected one of the five provided scenarios for your **application system design report (https://lms.griffith.edu.au/courses/24045/assignments/93487)** .