**Assignment 1: SDLC Overview - Create a one-page infographic that outlines the SDLC phases (Requirements, Design, Implementation, Testing, Deployment), highlighting the importance of each phase and how they interconnect.**

A) Software Development Life Cycle (SDLC) Overview*

---

*1. Requirements*

- *Description:* Gathering and documenting the needs and expectations of the stakeholders.
- *Importance:*
  - Defines the scope of the project.
  - Ensures alignment with business goals.
  - Prevents scope creep by having a clear and agreed-upon set of requirements.
- *Interconnections:*
  - *Feeds into Design:* Provides the blueprint for system architecture.
  - *Feedback Loop:* Ongoing updates and clarifications may be needed during the Design, Implementation, and Testing phases.

---

*2. Design*

- *Description:* Crafting the architecture of the system including detailed design specifications.
- *Importance:*
  - Lays out the system structure.
  - Ensures system scalability and reliability.
  - Provides a clear guide for developers.
- *Interconnections:*

  - *Informed by Requirements:* Directly based on the gathered requirements.

  - *Feeds into Implementation:* Provides detailed instructions for building the system.

---

*3. Implementation*

- *Description:* Actual coding and conversion of design into the software application.
- *Importance:*
  - Realizes the project plan into a functional product.
  - Iterative progress towards the final product.
  - Ensures adherence to design specifications.
- *Interconnections:*
  - *Driven by Design:* Follows the detailed plans from the design phase.
  - *Feeds into Testing:* Code needs to be verified through rigorous testing.

---

 *4. Testing*

- *Description:* Systematically checking the software for defects and verifying that it meets the requirements.
- *Importance:*
  - Ensures the software is bug-free.
  - Validates that the system performs as expected.
  - Improves software quality and user satisfaction.
- *Interconnections:*
  - *Based on Implementation:* Tests the code written during implementation.
  - *Feedback to Implementation:* Defects and issues found are reported back for fixing.

---

*5. Deployment*

- *Description:* Releasing the final product to users and making it operational.
- *Importance:*
  - Delivers the software to the end-users.
  - Ensures proper installation and configuration.
  - Training and support for users.
- *Interconnections:*
  - *Final step:* Utilizes the tested and validated software.
  - *Feedback Loop:* User feedback might prompt new requirements, starting the cycle anew.

---

## Assignment 2: Requirements Gathering - Conduct a 30-minute mock interview to gather requirements for a fictional app that helps organize community events. Summarize the requirements and how you would document and trace them in a one-page brief.

A) **Interview Summary Interviewee**: Sarah Johnson, Community Coordinator
**Date**: June 24, 2024
**Interviewer**: TEJA

**Objective**: To understand and gather requirements for an app that helps organize community events efficiently.

**Key Requirements**

1. **User Roles and Permissions**
   - **Admin**: Can create, manage, and delete events, and manage user roles.
   - **Event Organizer**: Can create and manage events, and view event reports.
   - **Participant**: Can view events, register for events, and receive notifications.
2. **Event Management**
   - **Create Events**: Users need to be able to input event title, description, date/time, location, category, and organizer details.
   - **Edit Events**: Users should be able to modify event details and cancel or postpone events.

    - **Event Templates**: Option to save templates for recurring events.
3. **Registration and Attendance**
    - **Registration Forms**: Customizable forms to gather specific information from participants.
    - **Attendance Tracking**: Features for check-in/check-out and real-time attendance tracking.
4. **Communication and Notifications**
    - **Email and SMS Notifications**: Automated reminders, updates, and post-event surveys.
    - **In-App Messaging**: Direct communication between organizers and participants.
5. **Calendar and Scheduling**
    - **Calendar Integration**: Sync events with Google Calendar, Outlook, etc.
    - **Scheduling Assistant**: Suggest optimal dates based on participant availability.
6. **Reporting and Analytics**
    - **Event Reports**: Generate reports on attendance, feedback, and finances.
    - **Analytics Dashboard**: Visual representation of event success metrics and participant demographics.
7. **Social Features**
    - **Event Sharing**: Share events on social media platforms.
    - **Community Feed**: Post event updates, photos, and community news.
8. **Payment Processing**
    - **Ticketing System**: Sell tickets, manage payments, and issue refunds.
    - **Sponsorship Management**: Track sponsorships and manage sponsor benefits.
9. **Accessibility and Support**
    - **Multi-Language Support**: The app should be available in multiple languages.
    - **Help Center**: Include FAQs, live chat support, and user guides.

**Documenting and Tracing Requirements**

1. **Requirements Document (RD)**
    - **Contents**: Includes introduction, scope, functional and non-functional requirements, user stories, use cases, and acceptance criteria.
    - **Tools**: Use Microsoft Word or Google Docs to create the document.
2. **Requirements Traceability Matrix (RTM)**
    - **Purpose**: Ensures every requirement is addressed during design, development, and testing.
    - **Attributes**: Include Requirement ID, description, priority, status, related user stories, and test cases.
    - **Tools**: Utilize Microsoft Excel or Jira for tracking.
3. **User Stories and Use Cases**
    - **Format**: Write user stories in the format "As a [user role], I want to [action] so that [goal]."
    - **Examples**:
        - As an Event Organizer, I want to create customizable registration forms so I can gather specific information from participants.

        - As a Participant, I want to receive SMS notifications about event updates so I stay informed.

4. **Acceptance Criteria**
    - **Definition**: Specific conditions to determine if a requirement is met.
    - **Examples**:
        - The event creation form must allow an organizer to input event details like title, description, and location.
        - The app must send a reminder email to participants 24 hours before the event.

**Implementation Plan**

1. **Initial Requirements Gathering**: Conduct interviews, surveys, and workshops with stakeholders.
2. **Documentation**: Compile information into the RD and RTM.
3. **Review and Validation**: Regular reviews with stakeholders to ensure accuracy.
4. **Prioritization**: Determine priority based on stakeholder input and feasibility.
5. **User Stories and Use Cases**: Develop detailed user stories and use cases.
6. **Acceptance Criteria**: Define clear acceptance criteria for each requirement.
7. **Continuous Traceability**: Maintain and update RTM throughout the project lifecycle.

---

## Assignment 3: Agile Principles Application - Write a two- paragraph reflection on how the Agile values of individuals and interactions, working solutions, and customer collaboration apply to the development of the community event app.

### A) Agile Principles Reflection

In developing the community event organizer app, the Agile value of prioritizing individuals and interactions over processes and tools plays a crucial role. Effective communication and collaboration among the development team, stakeholders, and end-users ensure that the app meets the actual needs of its users. Regular stand-up meetings, sprint reviews, and retrospectives facilitate ongoing dialogue and feedback, allowing the team to quickly address any issues and adapt to changing requirements. By focusing on these interactions, the team can create a more cohesive and motivated environment, ultimately leading to a more effective and user-centered product.

The Agile emphasis on working solutions over comprehensive documentation and customer collaboration over contract negotiation is also vital for this project. Rather than spending excessive time on detailed specifications and extensive documentation, the development team focuses on delivering functional increments of the app. This iterative approach allows for continuous testing and validation with real users, ensuring that the app evolves in alignment with user feedback and needs. Close collaboration with community coordinators and other stakeholders throughout the development process ensures that their insights and feedback are directly integrated into the app, leading to a product that truly supports the organization and management of community events. This customer-centric approach ensures the app remains relevant and valuable to its users.

---

## Assignment 4: Scrum Framework Overview - Prepare a one- page cheat sheet on the Scrum framework that includes roles, responsibilities, artifacts, and ceremonies. Provide a brief example of a Sprint task list for the earlier mentioned app project.

### A) Scrum Framework Cheat Sheet

*Roles and Responsibilities*

1. Product Owner
   - **Responsibilities**: Defines the product vision, manages the product backlog, prioritizes needs, and ensures the development team understands the requirements.
   - **Key Tasks**: Backlog refinement, stakeholder communication, and defining acceptance criteria.
2. Scrum Master
   - **Responsibilities**: Facilitates Scrum processes, removes impediments, ensures adherence to Scrum practices, and supports the team's productivity.
   - **Key Tasks**: Leading Scrum ceremonies, coaching the team, and removing blockers.
3. Development Team
   - **Responsibilities**: Delivers potentially shippable product increments, self-organizes, and commits to Sprint goals.
   - **Key Tasks**: Coding, testing, designing, and any other tasks needed to deliver the product increment.

*Artifacts*

1. Product Backlog
   - **Description**: An ordered list of everything that is known to be needed in the product.
   - **Owner**: Product Owner
   - **Components**: User stories, bug fixes, features, and technical work.
2. Sprint Backlog
   - **Description**: A list of tasks to be completed during the Sprint.

    - **Owner**: Development Team
    - **Components**: Selected Product Backlog items, task breakdown, and Sprint goals.
3. Increment
    - **Description**: The sum of all Product Backlog items completed during a Sprint and all previous Sprints.
    - **Owner**: Development Team
    - **Components**: Working software, integrated and potentially shippable.

### *Ceremonies*

1. Sprint Planning
    - **Purpose**: Determine what can be delivered in the upcoming Sprint and how it will be achieved.
    - **Participants**: Product Owner, Scrum Master, Development Team
2. Daily Stand-up (Daily Scrum)
    - **Purpose**: Synchronize activities and create a plan for the next 24 hours.
    - **Participants**: Development Team, Scrum Master
3. Sprint Review
    - **Purpose**: Inspect the Increment and adapt the Product Backlog if needed.
    - **Participants**: Product Owner, Scrum Master, Development Team, Stakeholders
4. Sprint Retrospective
    - **Purpose**: Reflect on the past Sprint and plan for improvements.
    - **Participants**: Scrum Master, Development Team

### *Example Sprint Task List for Community Event Organizer App*

1. User Registration Feature
    - Design registration form UI
    - Implement backend for user data storage
    - Integrate front-end with back-end
    - Conduct unit testing
    - Perform user acceptance testing (UAT)
2. Event Creation Module
    - Define event creation form fields
    - Implement event creation API
    - Develop front-end components for event creation
    - Conduct integration testing
    - Review with Product Owner
3. Notification System
    - Design email and SMS templates
    - Implement notification scheduling functionality
    - Integrate third-party notification service
    - Conduct system testing
    - Validate with sample data
4. Calendar Integration
    - Research calendar APIs (Google Calendar, Outlook)
    - Implement calendar sync feature
    - Develop UI for calendar integration settings
    - Perform end-to-end testing
    - Gather feedback from stakeholders