**Event Management Software Project**

Ikriti Karki, Sayushma Khadka, Kasish Shrestha

CSE 220 Principles of Software Engineering

International American university

Professor Chalise

August 25, 2025

**Gracious Designs - Complete Agile Development Documentation**

**Abstract**

The professional event management company, Gracious Designs, that deals with corporate and social event planning services is the owner of the project outcome, developed by our team. The implementation demonstrates how the principles of Agile methodology, more specifically the Scrum approach can be implemented practically to develop a web-based platform to manage events and also capable of meeting the genuine needs of clients. Full project lifecycle documentation to include requirement engineering, UML modeling when thinking through system design, sprint planning and implementation sections, and exhaustive testing methods are presented in the scholarly article. To create and develop the final product, our team utilized industry-standard development tools like GitHub to organize code sharing and version control, Trello as a project management leader, and a selection of UML visual editors to design the individual system architecture.

The firm was developed in response to the need of Gracious Designs to have a total digital solution to enhance efficiency in their operations, improve communication to clients, and ease its event planning processes. Some of the key deliverable are detailed Sprint 1 planning with roles and time-boxed tasks, a prioritized product backlog with ten user stories, well-documented functional and non-functional requirements, a stakeholder analysis list, and the four essential UML diagrams (Use Case, Class, Sequence, and Activity). Better collaboration, an iterative approach and continuous integration of client feedback were the results of the deployment of Agile. Along with providing students with real-life experience of working with the professional procedures of software development, this academic project is an effective step in demonstrating how the modern methods of software development can be utilized to come up with the scalable and user-friendly event management products.

**Table of Contents**

* [**Project Introduction**](#project-introduction)
* [**Company Overview**](#company-overview)
* [**Project Scope**](#project-scope)
* [**Team Roles**](#team-roles)
* [**Decor Backlog**](#product-backlog)
* [**Sprint 1 Planning**](#sprint-1-planning)
* **Gantt Chart- Sprint 1**
* [**Requirements Documentation**](#requirements-documentation)
* [**UML Diagrams**](#uml-diagrams)
* [**GitHub Repository Structure**](#github-repository-structure)
* [**Test Cases**](#test-cases)
* [**Agile Process Reflection**](#agile-process-reflection)

[**Project Introduction**](#project-introduction)

The goal of this academic project is to create a complete event management software system for the expert event management firm Gracious Designs. In order to improve customer communication, increase operational efficiency, and streamline Gracious Designs' event planning procedures, the student development team has been entrusted with developing a web-based application.

**Client Overview: Gracious Designs**

Gracious Designs is an established event management company that specializes in planning and executing corporate events, social gatherings, weddings, and large-scale celebrations. The company has identified the need for a comprehensive digital solution to replace their current manual processes and improve service delivery to their clients.

**Project Vision Statement**

To create a user-friendly, expandable event management system that meets the unique business requirements of Gracious Designs and showcases expert software development techniques and the application of Agile methodology in a learning environment.

**Proposed Technology Stack**

**Frontend**: React.js, Bootstrap CSS Framework

**Backend**: Node.js, Express.js

**Database**: MongoDB (NoSQL)

**Authentication**: JWT (JSON Web Tokens)

**Version Control**: Git/GitHub

**Project Management**: Trello

**Documentation**: Microsoft Office Suite

**Company Overview - Client Profile**

The client company for this academic project is Gracious Designs, a seasoned event management company that has been helping the community since 2018. Through painstaking preparation and faultless execution, the firm specializes in producing unforgettable experiences.

**Company Mission**

To create extraordinary events that exceed client expectations through innovative planning, attention to detail, and exceptional customer service.

**Core Services**

* **Corporate Event Planning**: Conferences, seminars, product launches, and team-building events
* **Social Event Management**: Weddings, anniversaries, birthday celebrations, and family gatherings
* **Venue Coordination**: Location scouting, booking, and management
* **Vendor Management**: Catering, entertainment, decoration, and technical service coordination
* **Budget Planning**: Cost estimation, expense tracking, and financial management
* **Timeline Management**: Event scheduling, milestone tracking, and deadline management

**Current Challenges (Client Requirements)**

* Manual client registration and profile management processes
* Inefficient event booking and availability tracking systems
* Complex vendor coordination requiring better contract management
* Time-consuming event timeline and task scheduling
* Difficult cost estimation and budget tracking
* Limited automated notification capabilities
* Manual report generation processes
* Lack of centralized client portal access

**Project Opportunity**

This academic project helps Gracious Designs update their processes through custom software development while offering a chance to apply classroom skills to real-world business difficulties.

**Project Scope**

**In-scope**

The provision of a user registration, authentication system is a part of the scope to ensure safety when giving access to the administrative, customer, attendee and vendor. Although, digital ticketing and guest registration systems will help by easing the process of onboarding a participant and controlling their access, an accessible event design and administration tool will make it a breeze to design and iterate the contents of an event. It would have a system of coordinating and managing vendors to ensure that vendors work and communicate effectively and there is proper supervision of the contracts. A real-time notification system will send out all parties involved alerts and reminders in a timely fashion. Additionally, secure and convenient financial operations will be ensured by integration of payment processing with websites such as Esewa or PhonePay. Also, this solution will be mobile-responsive to ensure cross-device access and a comprehensive analytics and reporting on different performance indicators dashboard. On the final note, administrators will be able to manage user accounts, permissions and control how an system functions which will be provided as an admin interface.

**Out of Scope (Future Phases)**

This time step of the project will concentrate on basic functions and some of the advanced functions will be planned to be included later in the stages of the project. Phase 2 will introduce native mobile applications on iOS and Android to increase accessibility and the mobile user experience beyond the mobile-responsive web version. In the same direction, AI-power event recommendations to individual experiences and support a multi-language option to serve international audiences are also proposed in Phase 2. Next will be the addition of live streaming integration capabilities to further enhance events and have the ability to provide both hybrid and virtual participation. Next, there is Phase 3, which will involve the advanced integration of CRM and give more access to client relationship management and enhance business knowledge.

**Team Roles**

|  |  |  |
| --- | --- | --- |
| **Team Member** | **Responsibility** | **Background** |
| **Kasish Shrestha** | Define requirements, prioritize backlog, client communication, stakeholder analysis. | Computer Science - Project Management Focus |
| **Ikriti Karki** | Facilitate ceremonies, remove impediments, coach team, documentation management. | Computer Science - Project Management Focus |
| **Sayushma Khadka** | System architecture design, technical documentation, UML diagrams, testing strategy. | Computer Science - Project Management Focus |

### Team Collaboration

### Since this is a planning and documentation-focused project, all team members collaborate on:

* Requirements gathering and analysis
* UML diagram creation and review
* Sprint planning and backlog management
* Test case development and documentation
* GitHub repository management
* Project documentation and reporting

**Décor Backlog**

## **Epic 1: User Management**

* **US001 – User Registration**  
  As a new user, I want to create an account so that I can access the event décor catalogue.
  + Story Points: **5**
  + Priority: **High**
  + Status: **To Do**
* **US002 – User Authentication**  
  As a registered user, I want to log in securely so that I can access my personal dashboard.
  + Story Points: **3**
  + Priority: **High**
  + Status: **To Do**
* **US003 – User Profile Management**  
  As a user, I want to update my profile information so that my details are current and accurate.
  + Story Points: **3**
  + Priority: **Medium**
  + Status: **To Do**

## **Epic 2: Event Management**

* **US004 – Event Creation**  
  As an event organizer, I want to create new events themes so that I can manage event details and logistics.
  + Story Points: **8**
  + Priority: **High**
  + Status: **To Do**
* **US005 – Event Listing and Search**  
  As an attendee, I want to search and browse events so that I can find events of interest.
  + Story Points: **5**
  + Priority: **High**
  + Status: **To Do**
* **US006 – Event Details View**  
  As an attendee, I want to view detailed event information so that I can make informed decisions about attendance.
  + Story Points: **3**
  + Priority: **Medium**
  + Status: **To Do**

## **Epic 3: Registration & Payment**

* **US007 – Event Registration**  
  As an attendee, I want to register for events so that I can secure my attendance.
  + Story Points: **5**
  + Priority: **High**
  + Status: **To Do**
* **US008 – Payment**  
  As an attendee, I want to do easily pay advance so that I don’t have confirmation
  + Story Points: **5**
  + Priority: **Medium**
  + Status: **To Do**

## **Epic 4: Communication**

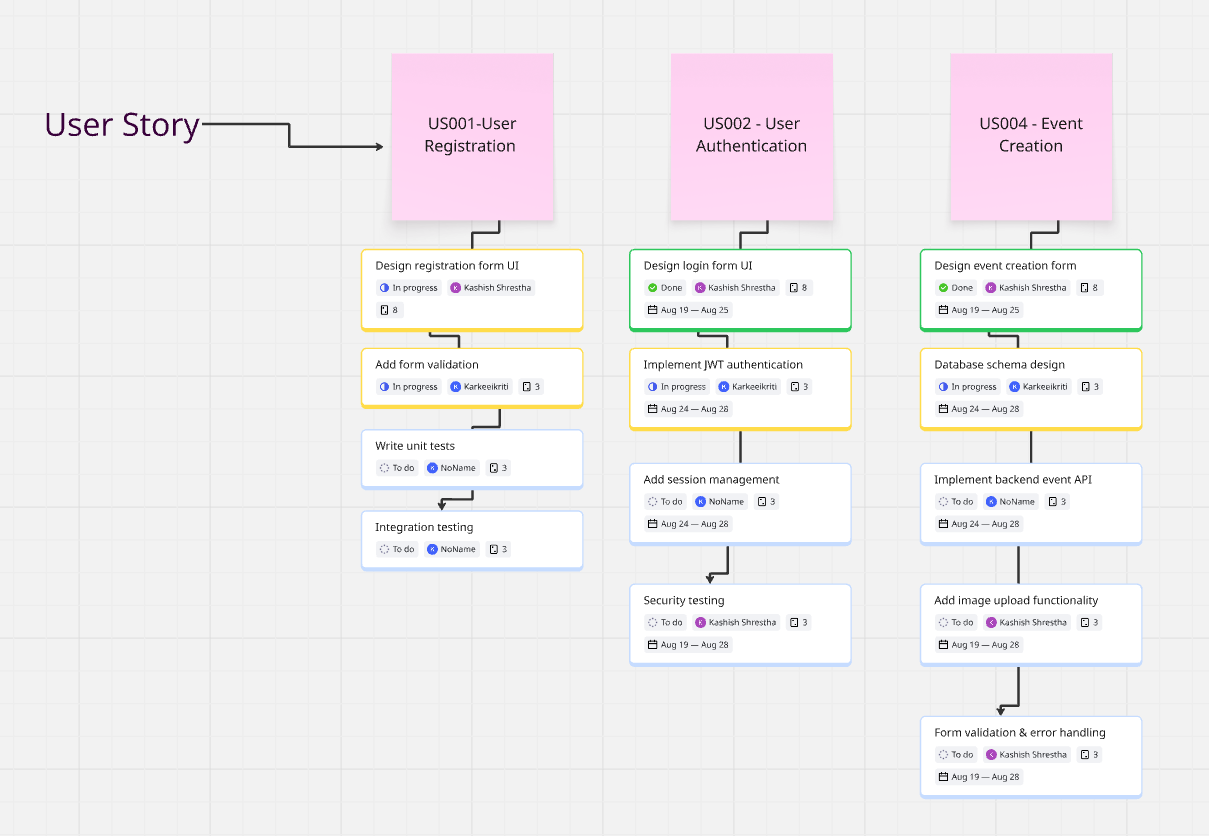
* **US009 – Notification System**  
  As a user, I want to receive notifications about events so that I stay informed about updates and changes.
  + Story Points: **8**
  + Priority: **Medium**
  + Status: **To Do**
* **US010 – Messaging System**  
  As an organizer, I want to communicate with attendees so that I can share important information.
  + Story Points: **8**
  + Priority: **Low**
  + Status: **To Do**

## **Sprint 1 Planning**

### Sprint Goal

Establish the foundation of the Event Management System with core user management and basic event creation functionality.

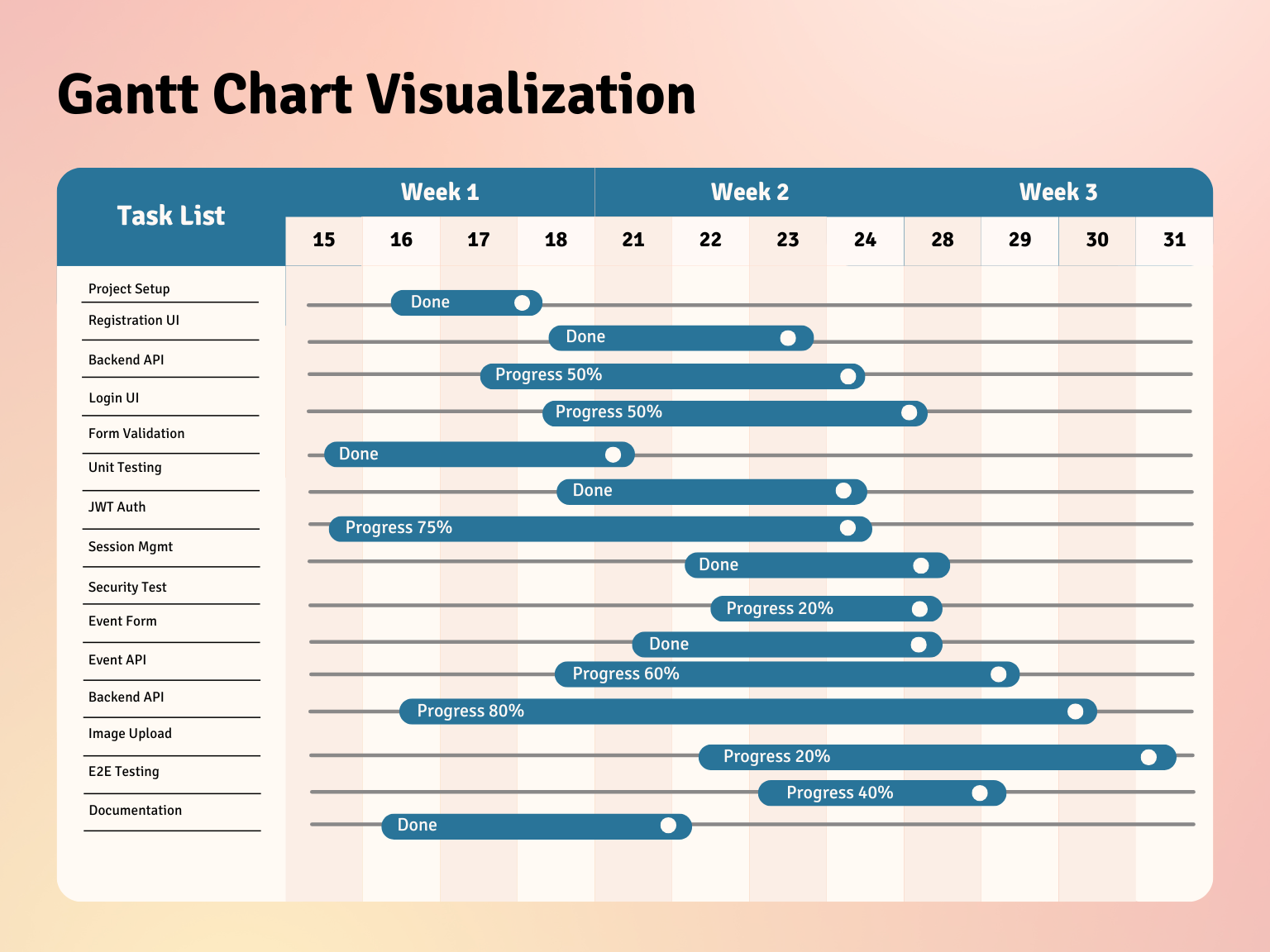
**Selected User Stories for Sprint 1**

****

<https://miro.com/app/board/uXjVJPm4WIM=/>

Using a visual a-board to attractively break user stories on the bigger picture into smaller more manageable tasks this image represents a well-organized sprint planning activity around your event management software project. US001 (User Registration), US002 (User Authentication), and US004 (Event Creation) are three key user stories that the board is composed of. The associated tasks are logged below each story with their assigned user, expected date to completion as well as the current status of those tasks (To-do, In progress, Done). In this detailed examination, the project team has a clear idea of the development process and is ensured that some specific and achievable tasks can be focused on during a particular sprint. The relationship between user story and their subtasks nourishes the logical flow and dependencies that is a quite helpful tool in clear tracking of the development progress and ensuring that the functional parts of each feature are covered.

**Gantt Chart - Sprint 1 Timeline**

****

During three weeks, the Gantt chart visualization provides an extensive overview of schedules and job progress in the project. Among some of the highlights are

Task Completion: Several tasks have been labeled as Done, indicating the notion of completion which includes Project Set-up, Registration UI, Unit Testing and Documentation.

Ongoing Progress: The percentages in the activities like the Backend API, Backend Login UI and Security Test, depict the current overall progress and in week one the Backend API is at 50%.

Task Management: This task management tool allows the interface to manage projects well through creation of task relationships and schedules. It also identifies areas that are seeking improvement such as the API and the Event Form that currently have lower percent complete.

Future Planning: Carefully monitoring the completion of these tasks in the following weeks, delays can be avoided and the project can be expected to remain on schedule because the available resources will have to be delegated accordingly.

Considering all of the above, Gantt chart proves to be an effective means of visualizing the project status and making decisions in time.

**Requirements Documentation**

**Functional Requirements**

**Client Registration and Profile Management:** Through the system, clients will be allowed to sign in by entering their business and personal data. A thorough profile of each client will be created which is inclusive of contact details, their preferences and a track record of events that the client has attended. Different access rights will also be given to different types of clients based on either individual, corporate or corner-based vendors.

**Venue Booking and Availability Tracking:** Real-time availability of venues across various areas will be displayed via the system. Customers will be able to look for venues by location, facilities offered, capacity, and date. The technology will allow customers to reserve locations while avoiding disputes or duplicate reservations. Additionally, it will maintain current calendars for every location, displaying available timeslots and booking status.

**Vendor Coordination and Contract Management:** A database of authorized vendors for various service categories will be maintained by the system. It will assist in allocating vendors to events according to the requirements. Terms and payment schedules for vendor contracts will be managed by the system. Additionally, it will monitor the performance of vendors and document customer feedback to support recommendations in the future.

**Event Timeline and Task Scheduling:** The system will track significant milestones and generate comprehensive event timelines. It will enable team members and vendors to be allocated tasks. For approaching deadlines, the system will automatically send out messages and reminders. Additionally, it will permit timeline modifications and demonstrate how they impact other activities and milestones.

**Cost Estimation and Budget Tracking:** Detailed cost estimates will be generated by the system according to each event's requirements. It will monitor the difference between approved budgets and actual spending. The technology will display any discrepancies between projected and actual expenses and offer real-time updates on the status of the budget. Additionally, it will manage several cost categories and assist with expense approval procedures.

**Automated Notification System:** Notifications for updates, reminders, and confirmations of reservations will be sent automatically via the system. It will enable a variety of communication modes, including as in-app messages, SMS, and email. Customizable notification templates for various event kinds will be possible with the system. Additionally, it will track the delivery status of all notifications and maintain logs of them.

**Report Generation Capabilities:** For events, the system will produce thorough reports that include financial summary. It will gather and aggregate comments and surveys related to client satisfaction. Reports on vendor performance and service quality will also be generated by the system. It will also provide bespoke reports with the ability to export and filter data as required.

**Client Portal Access:** Clients will have safe access to a customized gateway featuring dashboards of their own thanks to the system. Customers will be able to view the status of their budget, schedules, and events. The portal will facilitate approval workflows and permit document exchange. Additionally, it will allow clients to send feedback straight through the portal and interact with the team.

**Non-Functional Requirements**

**Performance:** 95% of inquiries will be answered by the system in less than two seconds. Up to 10,000 users will be able to use it simultaneously. It is anticipated that every database query will finish in 500 milliseconds.  
**Security:** To keep all passwords safe, the system will hash them using bcrypt. Appropriate authentication is necessary for API endpoints. Both the transmission and storage of payment information will be encrypted. In order to safeguard user information and privacy, the system will also adhere to GDPR regulations.  
**Usability:** The user interface of the system will function flawlessly on mobile devices, tablets, and PCs. It will be WCAG 2.1 AA compliant and accessible. When there are mistakes or missing data, all forms will display unambiguous messages to help users.

**Reliability:** With a 99.9% uptime, the system will be accessible virtually always. Every day, data will be backed up to guard against loss. The system will gradually deteriorate under heavy demand, preserving essential functionality without going totally down.

### Stakeholder Analysis

#### **Primary Stakeholders**

**Event Organizers**

* Interests: Easy event creation, attendee management, analytics
* Influence: High
* Requirements: Comprehensive event management tools

**Event Attendees**

* Interests: Easy registration, ticket management, event discovery
* Influence: High
* Requirements: Intuitive user interface, reliable ticketing

**System Administrators**

* Interests: System stability, security, maintainability
* Influence: Medium
* Requirements: Admin tools, monitoring capabilities

#### **Secondary Stakeholders**

**Vendors/Sponsors**

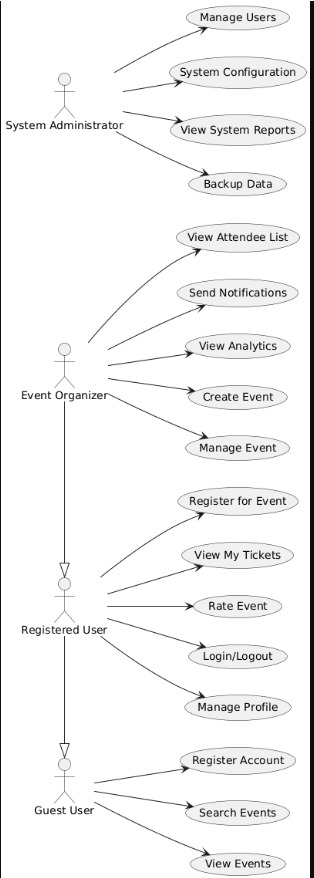
* Interests: Event promotion opportunities
* Influence: Medium
* Requirements: Vendor management features

**Payment Providers**

* Interests: Secure transactions, compliance
* Influence: Medium
* Requirements: API integration standard

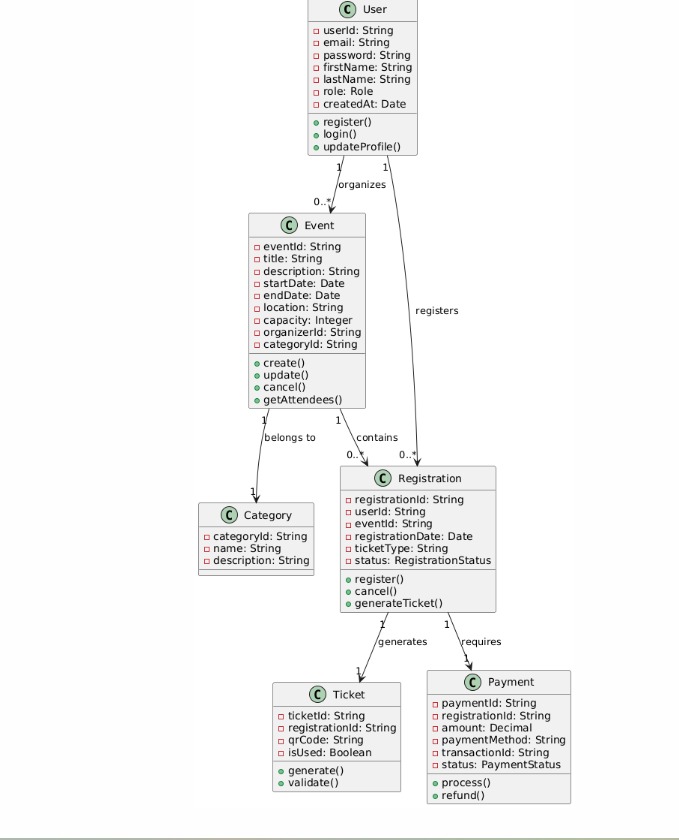
**UML Diagrams**

**Use Case Diagram**



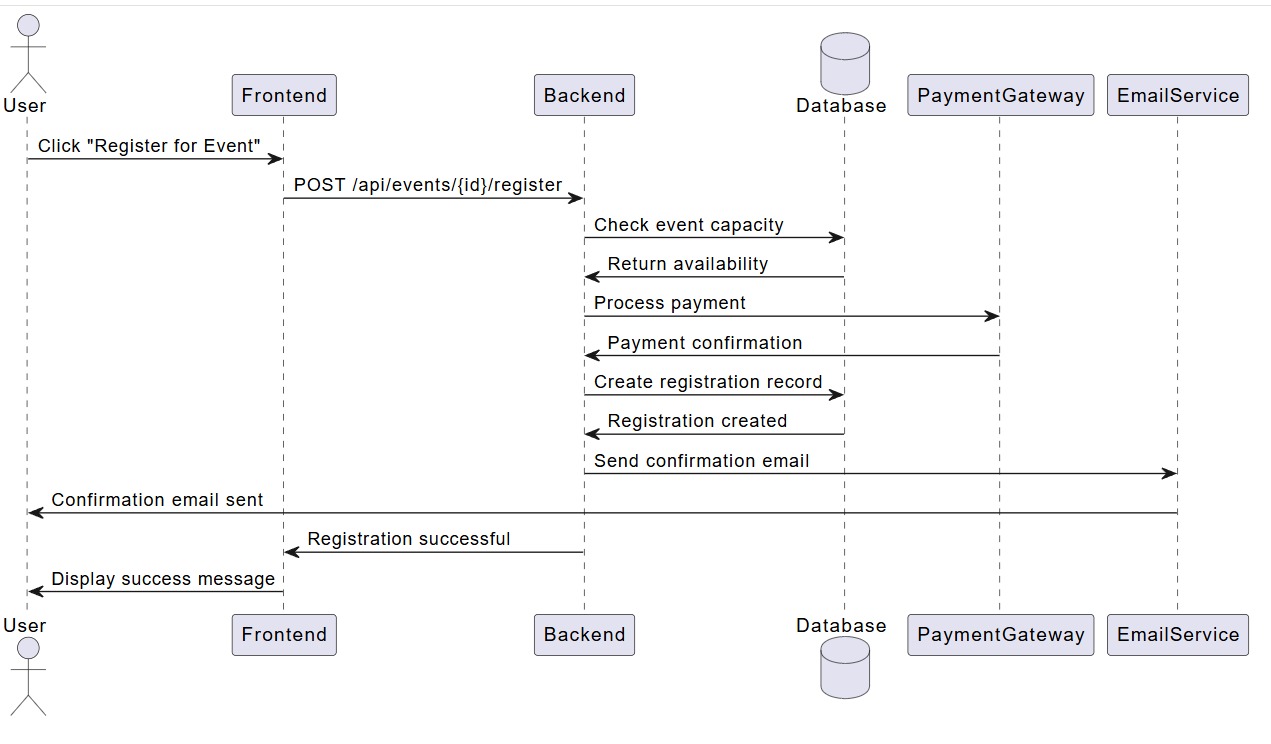
This is a use case diagram of the relations of different users and an event management system. The four principal players, the System Administrator, Event Organizer, Registered User and Guest User, are specified well, and have characteristic sets of use cases, or functions, that they can perform. The System Administrator, for example, will have complete access to all the high-level tasks such as: "Manage Users" and "System Configuration" whereas the Event Organizer will be able to perform the following tasks: Create, edit and notify events. However, there are differences between the Guest User and the Registered User in the sense that the former can only browse and view the events, whereas the latter is able to interact with the events online by registering, accessing their tickets, as well as rating events. The layout of the graphic effectively demonstrates the duties along with the prerogatives and responsibilities that attend them, and provides a neat overview of functional demands of the system as perceived by the user.

**Class Diagram**

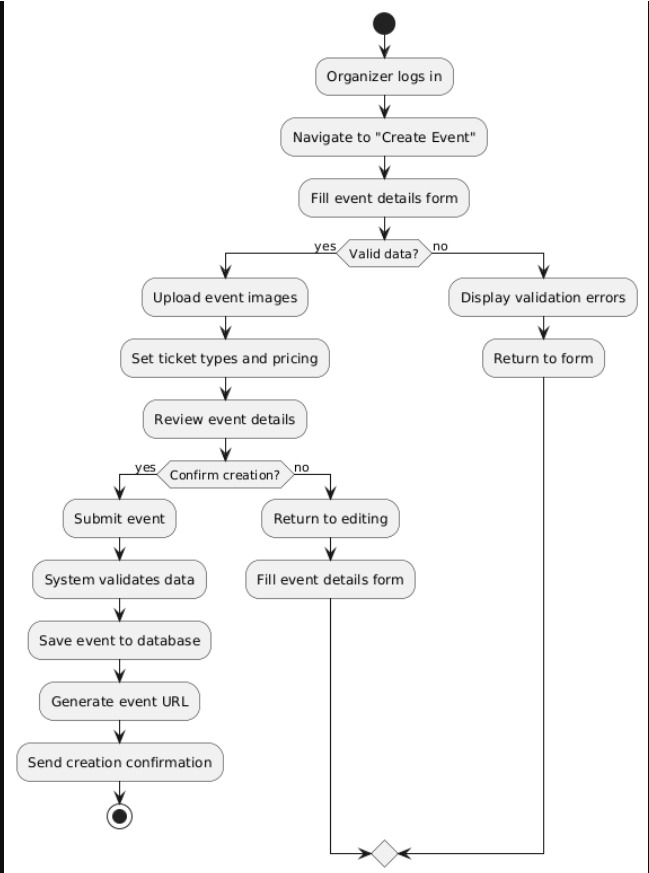


This describes how components are related and gives a visual representation of the structure of an event management system statically. In essence, it establishes many significant classes, with their own distinctive group of properties and methods, i.e. User, Event, and Registration. The user, in this case, has the behaviors and properties; register() and login() and email and role properties, respectively. The connections between the classes are represented by the lines that connect them, e.g. a user can create multiple events or register to multiple registrations. By specifying how many of these instances are involved in these interactions the multiplicity as in 1 and 0.\* supplies useful information of the architecture of the system. It is a template that should guide the development of the system and depicts the data model of the system, the fundamental relationships that exist amongst the highest ranking items of the system.

**Sequence Diagram - Event Registration**

This sequence diagram demonstrates how all the components of the system interact very precisely in a specific time sequence when a visitor registers to an event. When the user presses the button to register, the frontend sends the frontend to the backend, then the process begins. The Backend facilitates a series of these interactions e.g. a check of event capacity may require the use of the Database, a credit card payment to use the Payment Gateway and a record of the registration to be created. The Email Service sends the user confirmation of the saving of the record The stream of messages, the connection between one another (expect dependencies), and the step-by-step order in which the events occur are all well illustrated in the diagram finally ending with a successful registration message to be displayed on the frontend.

**Activity Diagram - Event Creation Process**



This activity diagram diagrammatically shows the step-by-step flow chart of procedures an event planner follows to create a new event. The initiator has to log in and go to the page of creating an event. A critical decision point of data validation is thus modelled with the diagram, that shows a forking route: upon actionable data, data streaming is to the next phase, say uploading photographs and setting ticket details; otherwise the user is shown issues and is redirected to the form. Before the final subset of events, the application sends the event to be validated, saves the changed data to the data base, develops a unique URL, and transfers a confirmation. This process consists of yet another option to confirm the creation. The visual representation of the event creation process is rather detailed as actions and decision nodes are used to produce a proper definition of the workflow logic, both conditional and sequential.

**GitHub Repository Structure**

## **Test Cases**

### Unit Testing

#### **Test Case: UC001 - User Registration**

* **Objective**: Verify user can register with valid credentials
* **Test Type**: Unit Test
* **Input**:
  + Email: "[test@example.com](mailto:test@example.com)"
  + Password: "SecurePass123!"
  + First Name: "Ram"
  + Last Name: "Shrestha"
* **Expected Output**: User created successfully, user ID returned
* **Actual Output**: [To be filled during execution]
* **Status**: Pass/Fail
* **Notes**: Tests password hashing and email validation

#### **Test Case: UC002 - User Registration Validation**

* **Objective**: Verify registration fails with invalid email
* **Test Type**: Unit Test
* **Input**:
  + Email: "invalid-email"
  + Password: "SecurePass123!"
* **Expected Output**: Validation error "Invalid email format"
* **Actual Output**: [To be filled during execution]
* **Status**: Pass/Fail

### Integration Testing

#### **Test Case: IC001 - Event Creation API**

* **Objective**: Test complete event creation workflow
* **Test Type**: Integration Test
* **Prerequisites**: User authenticated, valid token available
* **Input**:

json

{

"title": "Tech Conference 2025",

"description": "Annual technology conference",

"startDate": "2025-10-15T09:00:00Z",

"location": "Convention Center",

"capacity": 500

}

* **Expected Output**: Event created with unique ID, stored in database
* **Actual Output**: [To be filled during execution]
* **Status**: Pass/Fail

### System Testing

#### **Test Case: SC001 - End-to-End Event Registration**

* **Objective**: Complete user journey from registration to event attendance
* **Test Type**: System Test
* **Steps**:
  1. User creates account
  2. User searches for events
  3. User registers for event
  4. User receives confirmation email
  5. User views ticket in profile
* **Expected Output**: Successful completion of all steps
* **Actual Output**: [To be filled during execution]
* **Status**: Pass/Fail

### User Acceptance Testing

#### **Test Case: UAT001 - Event Organizer Workflow**

* **Objective**: Validate organizer can create and manage events effectively
* **Test Type**: User Acceptance Test
* **User Role**: Event Organizer
* **Scenario**: Create a new conference event with multiple ticket types
* **Acceptance Criteria**:
  + Event creation form is intuitive
  + All required fields are clearly marked
  + Image upload works correctly
  + Event appears in public listing
  + Organizer can view attendee list
* **Status**: Pass/Fail
* **User Feedback**: [To be collected during testing]

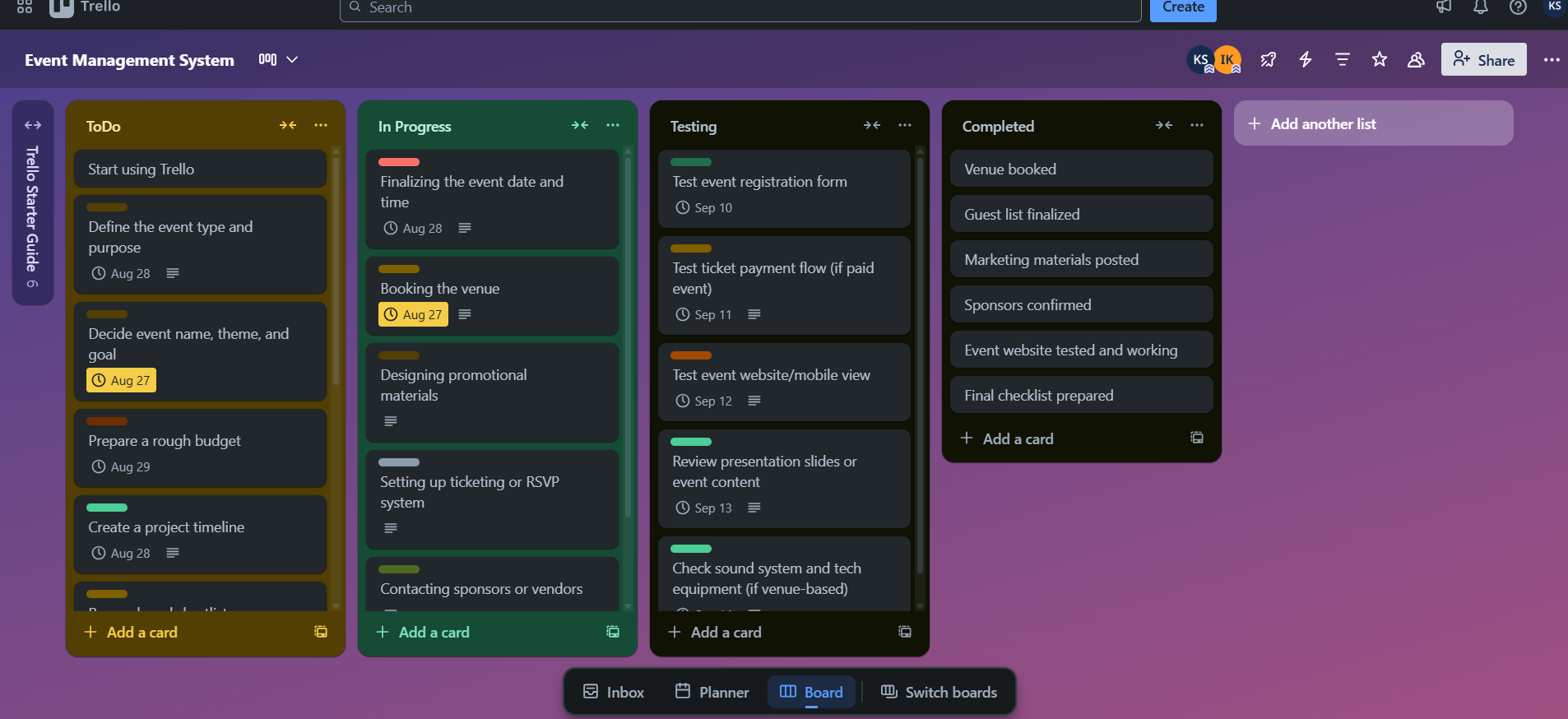
### Performance Testing

#### **Test Case: PT001 - Load Testing Event Listing**

* **Objective**: Verify system performance under load
* **Test Type**: Performance Test
* **Load**: 1000 concurrent users browsing events
* **Expected Response Time**: < 2 seconds
* **Actual Response Time**: [To be measured]
* **Status**: Pass/Fail

**Agile Process Reflection**

**Trello Board Organization**

****

In this picture, one can see a Trello board, which is a popular teamwork and project management tool. It is configured to host a Kanban-type workflow that tracks the progress of activities of a project, \"Event Management System\". The main column structure of the board consists of the following four columns: To Do, In Progress, Testing and Completed. Each task e.g. "Booking the venue" or "Test event registration form" are executed as card in each column. To ensure that the team is on track, the cards are containing deadlines in some of them. Due to the visual design, team members can easily and quickly understand what they have to do, what is in progress, and what they have already finished as this provides a clear and understandable image of the current state of the project. In such an arrangement, it is an excellent way of monitoring and organizing an event to the final stage.

**Conclusion**

Gracious Designs' Event Management Software project serves as an example of how Agile approaches may be successfully applied in a practical software development setting. Industry best practices in requirements engineering, system design, project management, and quality assurance are highlighted by the thorough documentation approach.

**Project Achievements**

With such a approach to the requirements analysis in both functional and non-functional areas and also stakeholder mapping, the project has demonstrated strong footing. Well-prioritized user stories, a properly structured product backlog, and clear acceptance criteria made it possible to have a strategic product planning. Suggestive effective sprint management were achieved through proper preparation, fair deadlines and clear roles. Some of the professional standards of documentation that have been followed are IEEE-aligned technical documentation and UML diagrams. Project implementation was further enhanced by deploying the modern development methods such as version control methods and collaborative efforts, enhanced quality assurance through the use of multi level testing processes to ensure coverage and verification.

**Gracious Designs' Methodology Impact**

The adoption of Agile technique of implementing this project is of great magnitude. Open communication and routine feedback of stakeholders resulted in enhanced collaboration with the clients. The need to maintain iterative value delivery and view progress was done through continuous software increments. The risk mitigation was conducted early by ensuring that issues are identified and resolved. Cross-functional collaboration and common responsibility of the outputs encouraged team empowerment. Lastly, the adaptive planning empowered the team to be flexible and be able to respond at high speed to the demands and market conditions.

**REFERENCES**

Pressman, R. S., & Maxim, B. R. (2020). *Software Engineering: A Practitioner's Approach*. McGraw Hill.

Sommerville, I. (2016). *Software Engineering* (10th ed.). Pearson.