|  |  |
| --- | --- |
| STUDENT NAME: | KAMOHO REUBEN NKUTU |
| STUDENT NUMBER: | ST10332693 |
| MODULE NAME: | PROGRAMMING |
| MODULE CODE: | PROG6212 |
| ACTIVITY TYPE: | PART ONE |

**Documentation:**

**Project Name**: The Contract Monthly Claim System

**Project Manager**: Kamoho Reuben Nkutu

**Overview:**

**Introduction**

The Contract Monthly Claim Systemis a .NET web-based application developed to streamline the process of submitting and approving monthly claims for independent contractor lecturers. By automating this process, CMCS aims to reduce manual errors, improve efficiency, and provide a centralized platform for managing contractor payments.

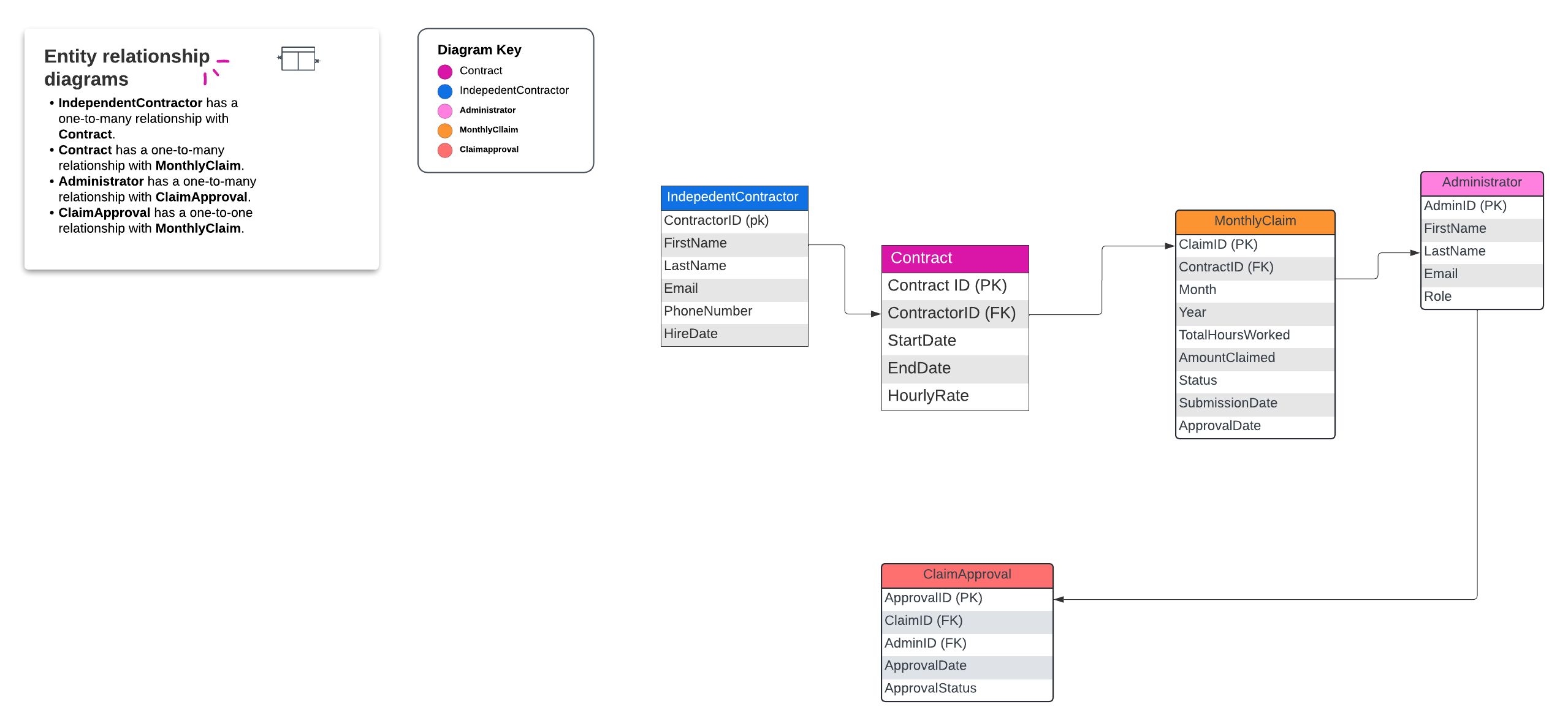
This documentation provides a comprehensive overview of the CMCS, including its features, setup instructions, and usage guidelines to help you successfully develop, deploy, and use the application.

**Features**

The Contract Monthly Claim System includes the following key features:

1. **User Authentication:**
   * Secure login for contractor lecturers and admin users.
   * Role-based access control to ensure that users can only perform actions relevant to their roles.
2. **Claims Submission:**
   * Simple and intuitive interface for contractor lecturers to submit their monthly claims.
   * Automated calculations for total claim amounts based on hours worked or tasks completed.
3. **Claims Approval:**
   * Admin users can view, approve, or reject submitted claims.
   * Automated notifications to contractors upon approval or rejection of their claims.
4. **Claims Management:**
   * Track the status of submitted claims in real-time.
   * View historical claims and their approval status.
5. **Reporting:**
   * Generate reports on approved claims, pending claims, and overall payment distributions.
   * Export reports in various formats for record-keeping.
6. **Email Notifications:**
   * Automatic email notifications to inform users about the status of their claims.
   * Configurable email settings to match your organization's SMTP server.
7. **Database Integration:**
   * Secure SQL Server database to store user information, claim details, and audit logs.
   * Database backup and restore functionality to ensure data integrity.

**UML Class Diagram:**



**Project Plan:**

**System Development Project Plan**

**1. Project Overview**

**1.1 Project Name**

Name: Contract Monthly Claim System (CMCS)

**1.2 Project Description**

Description: The Contract Monthly Claim System is a .NET web-based application that aims to streamline the process of submitting and approving monthly claims for independent contractor lecturers by automating the workflow.

**1.3 Project Objectives**

Objectives:

* Automate the submission and approval process for contractor lecturer claims.
* Reduce manual errors and increase processing efficiency.
* Provide a centralized platform for managing and tracking claims and payments.

**1.4 Project Scope**

* **In Scope:**
  + Development of the CMCS web application using .NET.
  + User authentication and role-based access control.
  + Claims submission, approval, and management functionalities.
  + Reporting and email notification features.
* **Out of Scope:**
  + Mobile application development.
  + Integration with third-party payroll systems.
  + Non-English language support.

**1.5 Deliverables**

Deliverables:

* CMCS web application with full functionality.
* User documentation and setup instructions.
* Technical documentation including database schema and API endpoints.
* Test cases and testing reports.

**1.6 Assumptions and Constraints**

* **Assumptions:**
  + All users will have access to a web browser compatible with the application.
  + The SQL Server database will be available and properly configured.
  + Stakeholders will provide timely feedback during the development process.
* **Constraints:**
  + The project must be completed within a 6-month timeframe.
  + The project budget is limited to R5 ,000,000.
  + The development must adhere to .NET framework guidelines and SQL Server compatibility.

**2. Project Organization**

**2.1 Stakeholders**

* Stakeholders:
  + Project Sponsor: Academic Affairs Department
  + End Users: Independent contractor lecturers, administrative staff
  + IT Department: For deployment and maintenance support

**2.2 Project Team**

* Project Manager: Kamoho Reuben Nkutu
* Team Members:
  + Lead Developer – Thabo William Smith
  + Front-end Developer – Johan de Kleek
  + Back-end Developer – Kabelo Sibusiso Dlamini
  + QA Engineer – Samuel Teboho Makhele
  + Database Administrator – Mapaseka Ngoma

**2.3 Roles and Responsibilities**

* **Roles and Responsibilities:**
  + **Project Manager:** Oversee project execution, manage resources, and communicate with stakeholders.
  + **Lead Developer:** Design system architecture and coordinate development tasks.
  + **Front-end Developer:** Develop the user interface and ensure a responsive design.
  + **Back-end Developer:** Implement business logic, manage server-side development.
  + **QA Engineer:** Conduct testing and ensure quality standards are met.
  + **Database Administrator:** Manage database design, implementation, and maintenance.

**3. Project Phases and Milestones**

**3.1 Initiation Phase**

* **Tasks:**
  + Define project goals and objectives.
  + Identify stakeholders.
  + Develop project charter.
* **Milestones:**
  + Project charter approval.

**3.2 Planning Phase**

* **Tasks:**
  + Develop project plan.
  + Define project scope.
  + Create work breakdown structure.
  + Develop schedule and budget.
  + Identify risks and create a risk management plan.
* **Milestones:**
  + Project plan approval.
  + Schedule and budget approval.

**3.3 Execution Phase**

* **Tasks:**
  + Develop system architecture and design.
  + Implement user authentication and role-based access control.
  + Develop claims submission and approval workflows.
* **Milestones:**
  + System architecture approval.
  + Completion of core functionalities (authentication, claims processing).
  + System integration complete.

**3.4 Monitoring and Controlling Phase**

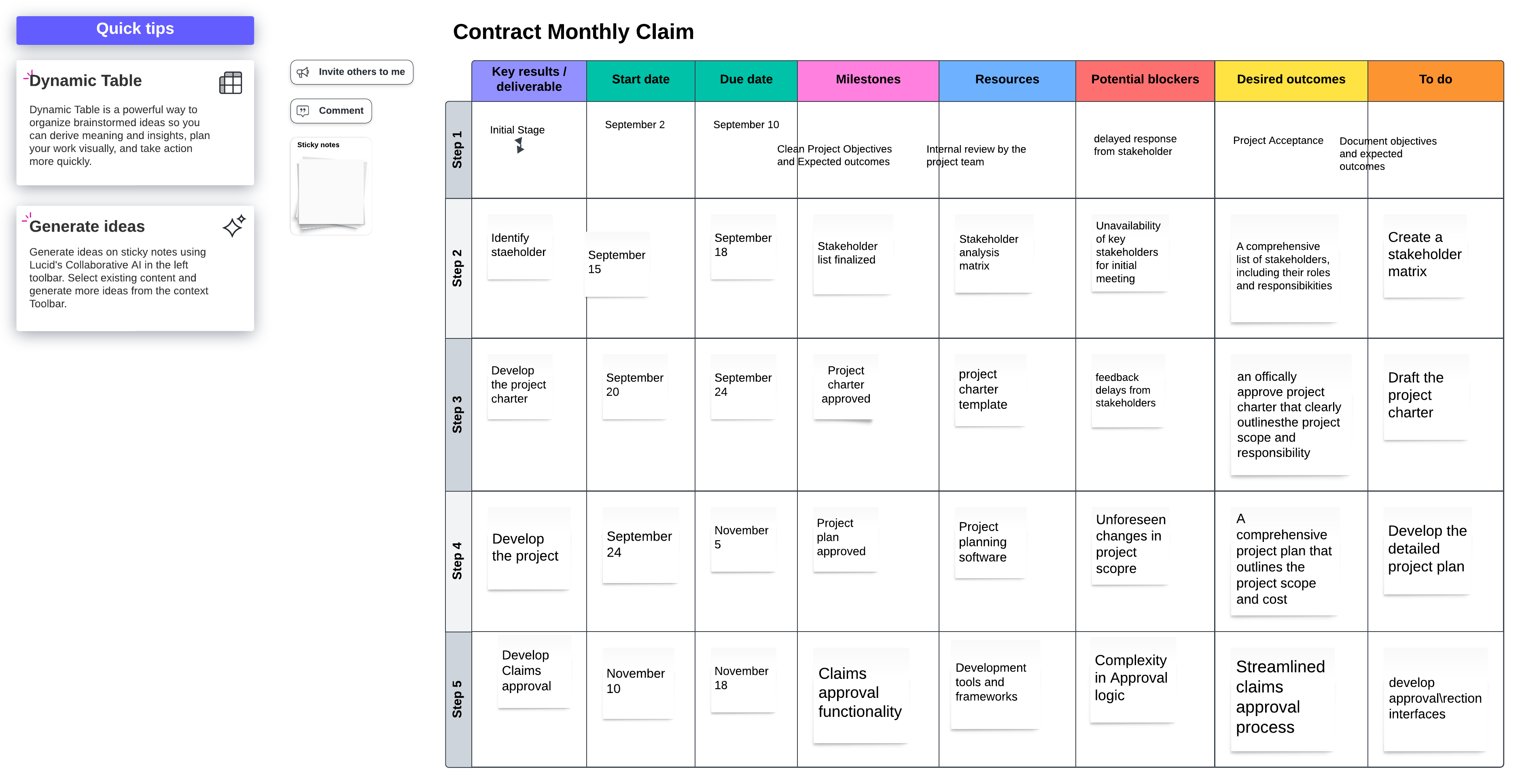
* **Tasks:**
  + Track project progress.
  + Perform quality assurance (QA) testing.
  + Manage changes and update project plan as needed.
  + Mitigate identified risks.
* **Milestones:**
  + Regular status reports.
  + Quality audits completion.

**3.5 Closure Phase**

* **Tasks:**
  + Conduct user acceptance testing.
  + Finalize documentation.
  + Deploy the CMCS application.
  + Conduct project review and gather feedback.
* **Milestones:**
  + UAT sign-off.
  + Project closure report.

**4. Project Schedule**

**4.1 Gantt Chart**



**4.2 Key Dates**

* Start Date: 1 September 2024
* End Date: 10 December 2024

**5. Budget and Resources**

**5.1 Budget**

* **Estimated Budget:** R500,000
  + Development Costs: R300,000
  + Testing and QA: R100,000
  + Deployment and Maintenance: R55,000
  + Miscellaneous Expenses: R55,000

**5.2 Resource Allocation**

* **Resources Required:**
  + Developer Workstations: 5
  + SQL Server Licenses: 1
  + Testing Tools: GitHub

**6. Risk Management**

**6.1 Risk Identification**

* **Risks:**
  + Delays in receiving stakeholder feedback.
  + Potential security vulnerabilities in user authentication.
  + Data loss due to database failure.

**6.2 Risk Mitigation Plan**

* **Mitigation Strategies:**
  + **Risk 1:** Schedule regular meetings with stakeholders to ensure timely feedback.
  + **Risk 2:** Conduct security audits and implement best practices for authentication.
  + **Risk 3:** Implement regular database backups and restore testing.

**7. Communication Plan**

**7.1 Communication Methods**

* **Methods:**
  + Meetings
  + Email updates
  + Status reports

**7.2 Frequency**

* **Frequency:**
  + Weekly status meetings.
  + Monthly progress reports.

**8. Quality Management Plan**

**8.1 Quality Objectives**

* **Objectives:**
  + Ensure the system meets all functional requirements.
  + Maintain high code quality and performance standards.

**8.2 Quality Assurance**

* **QA Activities:**
  + Code reviews
  + Testing

**9. Change Management Plan**

**9.1 Change Request Process**

* **Process:**
  + Submit a change request.
  + Review and approve change request.
  + Implement change.
  + Update project plan.

**10. Project Closure**

**10.1 Final Deliverables**

* **Deliverables:**
  + Final system release.
  + Documentation.

**10.2 Lessons Learned**

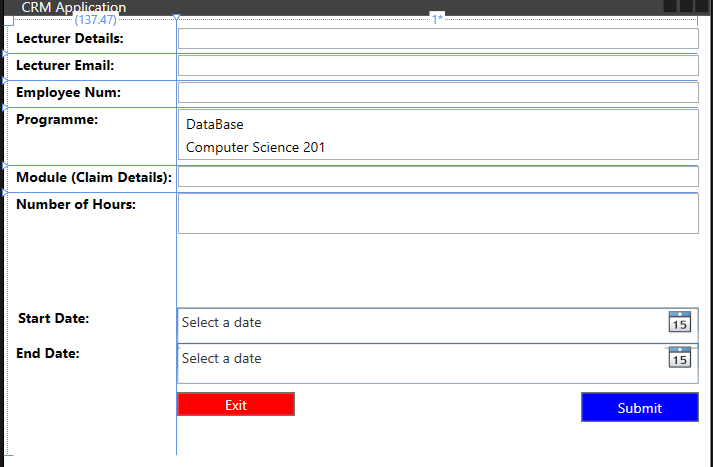
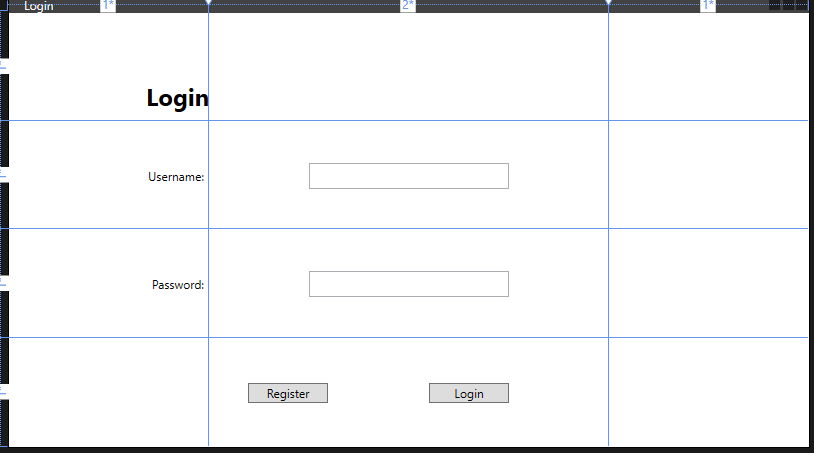
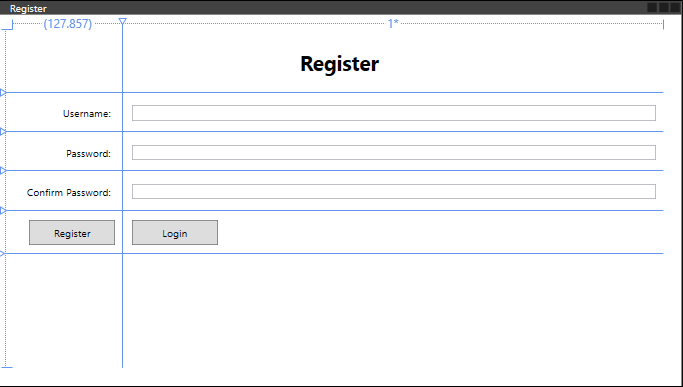
* **Lessons Learned:**
  + Importance of stakeholder engagement.
  + Need for rigorous security testing.
  + Value of regular backups and disaster recovery planning.

**10.3 Project Review**

* **Review Activities:**
  + Conduct projects debrief.
  + Archive project documents

|  |  |  |  |
| --- | --- | --- | --- |
| Milestone | Planned-Dates | Status | Notes |
| Project charter approval | Sep 1 – Sep 7 | Complete | Approved by stakeholders on time, |
| Project Plan approval | Sep 9 | Pending | On track, review meeting scheduled for Sep 9 |
| Schedule and Budget approval | Sep 15 – Sep 17 | On Track | Initial draft completed awaiting management approval |
| System Architecture approval | Sep 17 – Sep 28 | Not started | Scheduled to begin on Sep 17 |
| Completion of core functionalities | Sep 29 – Nov 9 | Not started | Development will start on Sep 29 |
| System Integration complete | Nov 13 - Nov 20 | Not started | Scheduled after development phase |
| Quality audits Completion | Nov 27 – Dec 5 | Not started | Planned after feature development |
| UAT sign-off | Dec 8 | Not Started | UAT planned for December |
| Project closure Report | Dec 10 | Not started | Final report pending after UAT |

GUI UI:



End.