**BURSARY APPLICATION MANAGEMENT SYSTEM**  
*By Christopher Gitau, Clinton Mboloi, Clive Misiko, Vincent Kimathi, Stephen Omondi & Gerishon Kihato*  
*Jomo Kenyatta University of Agriculture and Technology*  
*April 2025*

**CHAPTER ONE: INTRODUCTION**

**1.1 Background**

Globally, access to education is considered a fundamental right. Many developing countries, including Kenya, have put in place bursary programs to support students from financially disadvantaged backgrounds. However, the administration of these funds has remained a challenge, with most processes being manual, slow, and prone to errors.

In Kenya, county governments, institutions, and the national government provide bursaries to support students in secondary schools and tertiary institutions. Despite the noble intention, there is a growing concern about delays in processing, lack of transparency, misallocation of funds, and inadequate applicant communication. This calls for a digital transformation in bursary management.

**1.2 Statement of the Problem**

The current bursary management process in most Kenyan institutions is predominantly manual. This introduces several problems:

* **Inefficiency**: Paper‑based forms often get misplaced or incorrectly filled.
* **Lack of transparency**: Applicants receive little to no feedback on the status of their applications.
* **Delayed disbursement**: Approval cycles are lengthy due to lack of automation and poor tracking systems.

These challenges contribute to low satisfaction among applicants and hinder timely access to financial aid.

**1.3 Proposed Solution**

This research proposes the development of a **web‑based Bursary Application Management System** that allows students to apply for bursaries online, upload necessary documents, and track their application status. The system will enable administrators to vet, approve, and disburse bursaries more efficiently.

**1.4 Objectives**

**General Objective**

To develop a digital Bursary Application Management System that streamlines application, vetting, and communication processes for Kenyan institutions.

**Specific Objectives**

1. To enable online bursary application submissions and document uploads.
2. To implement role‑based access control for applicants and administrators.
3. To provide automated notifications and real‑time status updates to applicants.
4. To evaluate the system’s effectiveness in reducing processing time and improving transparency.

**1.5 Research Questions**

1. What challenges are faced in the current manual bursary application processes?
2. How can a web‑based system improve transparency and efficiency in bursary allocation?
3. What technologies and frameworks are most suitable for developing the system?
4. How will users interact with and benefit from the system functionalities?

**1.6 Justification**

This research aims to support Kenya’s Vision 2030 and Sustainable Development Goal 4 by promoting equitable access to quality education. A digitized bursary system can reduce paperwork, ensure fairer access to funds, and serve as a case study for broader digital governance initiatives in the education sector.

**1.7 Proposed Research and System Methodologies**

The research will follow a mixed‑methods approach, combining qualitative interviews with bursary administrators and quantitative surveys targeting applicants. For system development, the Agile methodology will be used to allow iterative development and stakeholder feedback. The MVC (Model‑View‑Controller) architecture will guide the system structure.

**1.8 Scope**

This study will focus on bursary schemes in Kenyan universities and county programs. It will not integrate with national financial bodies such as HELB but may lay the groundwork for such integrations in future versions.

**CHAPTER TWO: LITERATURE REVIEW**

**2.1 Introduction**

This chapter reviews literature related to digital bursary management systems, focusing on global best practices, relevant theories, technologies used, and empirical outcomes. It highlights the need for automation and transparency in public financial aid management.

**2.2 Theoretical Review**

* **Digital Governance Theory**: ICT enhances public service delivery, builds trust, and reduces corruption.
* **Agile Development Theory**: Flexibility and incremental delivery improve alignment with stakeholder needs.
* **Equity Theory in Education**: Need‑based resource allocation ensures all students have equal opportunities.

**2.3 Empirical Review**

* **EdTech Hub (2023)**: Manual bursary systems increase processing time by 40%.
* **UNESCO (2022)**: Rwanda’s digital bursary platform led to a 60% improvement in approval speed and applicant satisfaction.
* In Kenya, some counties have piloted partial digitization, but most still rely on manual vetting and paperwork.

**2.4 Summary and Research Gaps**

Despite global successes, there is limited research on implementing digital bursary systems in low‑resource, rural Kenyan contexts. Additionally, few studies address user‑experience challenges for applicants with limited digital literacy.

**CHAPTER THREE: METHODOLOGY**

**3.1 Research Design**

This study adopts a mixed‑methods approach:

* **Quantitative**: Surveys (n=100) to capture applicant challenges and expectations.
* **Qualitative**: Interviews (n=10) with bursary officers to map workflows and requirements.

**3.2 Feasibility Study**

* **Technical Feasibility**: Use of open‑source technologies (React, Node.js, MySQL) reduces cost.
* **Operational Feasibility**: Most users are familiar with smartphones and web applications.
* **Time Feasibility**: The project is planned over an 8‑week cycle from design through deployment.

**3.3 Data Collection Tools and Requirements Elicitation**

* **Structured Questionnaires**: Gather quantitative data on applicant pain points.
* **Stakeholder Interviews**: Qualitative insights from bursary administrators.
* **Observation**: Direct review of existing manual processes to identify bottlenecks.

**3.4 Data Analysis and System Requirements Specification**

Collected data will be analyzed using Excel. Bar and pie charts will visualize common issues and priorities. Findings will inform the system’s functional requirements, including:

* **User Roles** (Applicant, Administrator)
* **Data Elements** (personal details, income info, academic records)
* **Notification Triggers** (submission, approval, rejection)

**3.5 System Architecture and Design Overview**

The system follows an MVC pattern:

* **Model**: MySQL database stores users, applications, and logs.
* **View**: React.js delivers responsive interfaces.
* **Controller**: Node.js/Express.js handles business logic, validation, and API endpoints.

**3.6 Design**

**3.6.1 General Overview of the System**

The new system replaces the CDF’s paper‑based process—where application windows are posted on notice boards and feedback is uncertain—with an online portal that offers clear timelines and status updates.

**Figure 1:** General Overview of the System

**3.6.2 Data Flow Diagrams**

* **3.6.2.1 Context Diagram**

**Figure 2:** Context Diagram

* **3.6.2.2 Level 0 DFD**

**Figure 3:** Level 0 DFD

* **3.6.2.3 Level 1 DFD**

**Figure 4:** Level 1 DFD

**3.7 Use Case Diagram**

Actors and interactions are defined to capture all user stories.

**Figure 5:** Use Case Diagram

**3.8 Activity Diagram**

Describes dynamic workflows for both applicants (register, submit, track) and administrators (login, review, update).

**Figure 6:** Activity Diagram

**3.9 Class Diagram**

Illustrates core classes—Applicant, Application, Administrator, Disbursement—and their attributes and methods.

**Figure 7:** Class Diagram

**3.10 ERD Diagram**

Defines entities (Applicant, Application, Document, Disbursement) and relationships.

**Figure 8:** ERD Diagram

**CHAPTER FOUR: SYSTEM ANALYSIS, DESIGN AND TESTING**

**4.1 Introduction**

This chapter analyzes the system’s requirements in detail, outlines the design decisions referencing Chapter 3 models, and presents the comprehensive testing strategy used to validate functionality, performance, and usability.

**4.2 Functional Requirements**

1. **User Registration & Authentication**
2. **Bursary Application Submission** (with document upload)
3. **Real‑Time Status Tracking**
4. **Administrator Application Management** (view, vet, approve/reject)
5. **Automated Notifications** (email/SMS upon status changes)

**4.3 Non‑Functional Requirements**

* **Usability**: Intuitive UI, responsive across devices.
* **Security**: HTTPS, JWT authentication, encrypted passwords.
* **Scalability**: Modular design to onboard multiple institutions.
* **Maintainability**: Clean codebase, documented APIs, MVC separation.

**4.4 System Architecture**

Reiterating the MVC setup:

* **Model Layer**: MySQL schemas for users, applications, logs.
* **View Layer**: React.js components for forms, dashboards, notifications.
* **Controller Layer**: Express.js routes handling CRUD operations, validation, and business rules.

**4.5 System Design Tools Used**

* **DFDs** (Context, Level 0, Level 1)
* **Use Case Diagrams**
* **Activity Diagrams**
* **Class Diagrams**
* **ERD Diagram**

*Note: All UML and data‑flow diagrams appear in Chapter 3; they are referenced here and not repeated in full.*

**4.6 System Testing**

**4.6.1 Testing Approach**

| **Test Type** | **Purpose** |
| --- | --- |
| **Unit Testing** | Verify individual functions/modules (form validation, etc.) |
| **Integration Testing** | Ensure modules (UI ↔ API ↔ DB) interact correctly |
| **System Testing** | End‑to‑end workflows under expected load |
| **User Acceptance Testing** | Validate usability and requirements with real users |

**4.6.2 Sample Test Cases**

| **TC ID** | **Description** | **Expected Result** | **Outcome** |
| --- | --- | --- | --- |
| TC001 | Login with valid credentials | Redirect to dashboard | Passed |
| TC002 | Submit application without required docs | Display “Please upload all required docs.” | Passed |
| TC003 | Admin approves application | Status updates to “Approved” immediately | Passed |
| TC004 | Notification sent upon status change | SMS/email received by applicant | Passed |

**4.6.3 Tools Used for Testing**

* **Postman**: API endpoint verification
* **Jest**: Automated unit tests for backend logic
* **Manual/UAT**: Hands‑on testing with students and admins
* **Feedback Forms**: Structured surveys during UAT to capture usability issues

**4.6.4 Testing Challenges**

* **SMS API Rate Limits**: Occasional delays in notifications.
* **File‑Upload Size**: Required adjusting front‑end limits and adding validation messages.
* **Mobile Responsiveness**: Initial layouts broke on smaller screens—iterated CSS and component structure.

**End of Chapters 1–4**