|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **SPRINT** | **TIMELINE** | **GOAL** | **KEY TASK** | **FRONT-END DEVELOPER RESPONSIBILITIES** | **BACK-END DEVELOPER RESPONSIBILITIES** | **QUALITY ASSURANCE TESTER RESPONSIBILITIES** | **PROJECT MANAGER RESPONSIBILITIES** | **EXPECTED OUTCOME (MILESTONE)** | **VALUE VS. EFFORT** |
| 1 | Weeks  1-3 | System Setup & User Management | ✅ Setup project repository and development environment  ✅ Design database schema for users and authentication  ✅ Implement User Registration & Login (RBAC, Password Recovery)  ✅ Develop Question Bank System (Storage, Categorization, Search)  ✅ Build Student Dashboard (Performance Overview, Test Results)  ✅ Initial UI Design and Navigation Structure | 🔹 Develop **Authentication UI** (Login, Registration, Forgot Password)  🔹 Build **Student Dashboard UI** (Performance Overview, Test History, Summary Cards)  🔹 Implement **Navigation & Responsiveness** (Mobile-first approach)  🔹 Connect frontend to authentication API and Question Bank API | 🔹 Design **Database Schema** (Users, Roles, Authentication, Question Bank)  🔹 Implement **RBAC System & Authentication API**  🔹 Develop **Question Bank API (CRUD, Search, Filters)**  🔹 Optimize **Database Queries for Performance** | 🔹 Test **User Authentication** (Login, Registration, Password Recovery)  🔹 Validate **Navigation & UI Flow**  🔹 Verify **API and Database Structure**  🔹 Document any UI/UX inconsistencies | 🔹 Oversee **Sprint Execution**  🔹 Ensure **Database & API align with frontend development**  🔹 Conduct **Stakeholder Review for UI & UX Feedback**  🔹 Monitor sprint progress and resolve blockers | 🎯 Users can **log in, register, and recover passwords**.  🎯 Faculty can **store and retrieve questions in the Question Bank**.  🎯 Students can **view their test history and performance metrics**. | High Value –  Low Effort |
| 2 | Weeks  4-5 | Test Management & Auto-Grading | ✅ Develop Basic Test Creation Module  ✅ Implement Automated Scoring for Objective Questions (MCQs)  ✅ Enable Question Printing System  ✅ Implement Basic Question Randomization  ✅ Optimize test retrieval & performance | 🔹 Build **Test Creation UI**  🔹 Implement **Auto-Grading UI** (Display Correct/Incorrect Answers & Scores)  🔹 Design **Print-Friendly Test Layouts** (PDF Export, Print Preview)  🔹 Optimize **Test-Taking UI for students** | 🔹 Develop **Test Creation API** (Faculty creates & manages tests)  🔹 Implement **Auto-Scoring Algorithm**  🔹 Improve **Test Randomization & Performance** | 🔹 Test **Test Creation Workflow**  🔹 Validate **Auto-Grading Accuracy**  🔹 Ensure **Correct PDF Format for Printing**  🔹 Check **Randomization Logic** | 🔹 Ensure **Test Module aligns with faculty needs**  🔹 Gather **feedback from faculty testers**  🔹 Monitor **Performance Optimization** | 🎯 Faculty can **create tests, randomize questions, and print test papers**.  🎯 Students can **take quizzes and get automated scores**. | **High Value**  **– Medium Effort** |
| 3 | Weeks  6-7 | Approval System & Secure Testing | ✅ Develop Approval System for Faculty Updates  ✅ Implement Secure Online Testing (Anti-cheating, Full-Screen Lock)  ✅ Create Tagged Question Removal System  ✅ Generate Statistical Analysis Reports | 🔹 Develop **Approval System UI** (Pending, Approved, Rejected Questions)  🔹 Implement **Secure Test-Taking UI** (Full-screen mode, Timer, Anti-Cheating Measures)  🔹 Build **Reports Dashboard UI** (Faculty sees question performance statistics) | 🔹 Implement **Approval System Logic** (Faculty submits, Program Chair & Dean approve changes)  🔹 Develop **Anti-Cheating Features** (Restrict tab switching, IP tracking)  🔹 Implement **Tagged Question Removal System** (Flag poorly performing questions)  🔹 Create **API for Statistical Reports** | 🔹 Test **Approval System Workflow**  🔹 Validate **Security of Online Testing**  🔹 Verify **Statistical Reports Accuracy** | 🔹 Coordinate **faculty & admin feedback on approval system**  🔹 Ensure **secure testing meets compliance and security standards** | 🎯 Faculty can **submit, review, and approve question changes**.  🎯 Secure **online tests prevent cheating**.  🎯 Tagged **question reports improve question bank quality**. | High Value –  High Effort |
| 4 | Weeks  8-9 | Faculty Control & Randomization | ✅ Implement Faculty Assignment & Reassignment System  ✅ Develop Subject & Difficulty Selection Control  ✅ Improve Advanced Question Randomization  ✅ Enable System Notifications | 🔹 Build **Faculty Management UI** (Assign roles, reassign faculty)  🔹 Design **Subject & Difficulty Control UI**  🔹 Improve **Question Selection Interface** (Advanced randomization settings)  🔹 Implement **System Notifications UI** | 🔹 Develop **Faculty Role Management System**  🔹 Implement **Advanced Question Randomization**  🔹 Enable **System Notifications API** (Push alerts for approvals, updates) | 🔹 Test **Faculty Role Assignments & Modifications**  🔹 Validate **Question Randomization Algorithm** | 🔹 Ensure **faculty control system is user-friendly**  🔹 Validate **notification effectiveness** | 🎯 Super Admins can **manage faculty roles and subject difficulty**.  🎯 Faculty and students receive **notifications for approvals and updates** | Low Value –  Low Effort |
| 5 | Weeks 10-11 | UI Enhancements & Campus-Specific Features | ✅ Implement Theme Customization (Light/Dark Mode)  ✅ Develop Gamification Elements (Badges, Leaderboards)  ✅ Restrict access via Campus-Specific Access Control  ✅ Create Inter-Campus Performance Dashboard  ✅ Integrate JRSMSU Email for Login | 🔹 Implement **Theme Switcher**  🔹 Build **Gamification UI**  🔹 Develop **Campus-Specific Access UI**  🔹 Design **Inter-Campus Dashboard** | 🔹 Develop **Campus-Based Features API**  🔹 Implement **JRSMSU Email Restriction** | 🔹 Test **Campus-Specific Access Restrictions**  🔹 Validate **Gamification Functionality** | 🔹 Ensure **UI improvements align with user needs**  🔹 Monitor feedback on campus-specific restrictions | 🎯 Users can **switch themes**.  🎯 Gamification encourages student participation.  🎯 Campus-based access enhances security. | Low Value –  High Effort |
| DEPLOY  MENT | Week 12 | System Testing & Launch | ✅ Conduct User Testing  ✅ Perform Bug Fixes & Optimization  ✅ Obtain Final Approval  ✅ Deploy System | 🔹 Final **UI/UX Testing**  🔹 Fix **UI Issues**  🔹 Implement **Onboarding Tooltips** | 🔹 Optimize **Performance & Security**  🔹 Deploy **System to Live Server** | 🔹 Conduct **Final Testing & Security Checks** | 🔹 Oversee **Deployment & Sign-Off**  🔹 Gather **Final Feedback** | 🎯 **Fully deployed system**, ready for use 🎉 | High Value –  High Effort |