**School Management system**

**Overview**

Abram's School Management System is a comprehensive web application designed to manage various aspects of school administration, including user management, communication, class management, and academic activities. The system is developed using the Waterfall Model, a traditional software development methodology that follows a linear and sequential approach. This document outlines the seven stages of the Waterfall Model and details the features of the Abram's School Management System.

**1. Requirement Analysis**

During the Requirement Analysis phase, we gathered all the necessary information to define the functionalities and features of Abram's School Management System. The requirements were collected through meetings with stakeholders, including school administrators, teachers, students, and IT professionals.

**Key Features Identified:**

1. **Login and Signup with Image Upload**: Users, including students, teachers, and admins, can create accounts and upload profile pictures during registration. Login functionality is secured with authentication methods.
2. **Contact with School**: A contact form allows anyone to reach out to the school administration by providing their name, email, and message.
3. **Apply to the School**: Prospective students can apply to the school by filling out an application form, which includes personal details and supporting documents.
4. **Search for Application and Results**: Users can search for their application status and view the results of their application.
5. **Admin and Teacher Dashboard**: A powerful dashboard for admins and teachers, featuring an overview of school activities, user management, and academic data.
6. **Teacher Management by Admin**: Admins can create teacher accounts, view the list of available teachers, manage the registration queue, and change the status of applications.
7. **Classroom Management by Teachers**: Teachers can create virtual rooms, view rooms they've created, and upload materials and assignments.
8. **Student Participation**: Students can join rooms, access materials, complete assignments, and participate in discussions by commenting on posts.
9. **User Logout**: Secure logout functionality for all user roles.
10. **Security Features**: Strong security mechanisms, including token-based authentication with refresh tokens, are implemented to protect user data and system integrity.

**2. System Design**

In the System Design phase, the architectural design of Abram's School Management System was created based on the requirements gathered. This phase involved creating diagrams, models, and schemas to provide a visual representation of the system's structure.

**Design Elements:**

1. **User Interface Design**: The user interface was designed with simplicity and usability in mind. Key pages include login/signup forms, dashboards, contact forms, and classroom interfaces.
2. **Database Design**: The database schema was designed to efficiently store user data, application information, classroom details, and comments. Tables were created for users, applications, classrooms, materials, assignments, and more.
3. **Security Design**: A robust security framework was designed, incorporating token-based authentication, secure password storage, and user role management (admin, teacher, student).
4. **Dashboard Design**: The admin and teacher dashboards were designed to display relevant data at a glance, with functionalities like teacher creation, room management, and queue management.

**3. Implementation**

The Implementation phase involved coding and integrating all the components designed in the previous phase. This phase was divided into smaller tasks, with each feature being implemented separately.

**Implementation of Key Features:**

1. **Login and Signup with Image Upload**: User authentication was implemented using JWT (JSON Web Tokens) with support for image uploads during signup.
2. **Contact with School**: A contact form was implemented using React for the frontend and Node.js for backend handling, storing inquiries in the database.
3. **Apply to the School**: A student application form was created, allowing users to submit personal details and documents. The backend processes and stores the application data.
4. **Search for Application and Results**: A search feature was implemented to allow users to query their application status and results.
5. **Admin and Teacher Dashboard**: The dashboards were implemented using React and integrated with the backend to fetch and display data dynamically.
6. **Teacher Management by Admin**: Admin functionalities were implemented to manage teacher creation, view lists, and handle registration queues.
7. **Classroom Management by Teachers**: Teachers can create and manage virtual classrooms, upload materials, and assign homework using the dashboard.
8. **Student Participation**: Students can join classrooms, access materials, and comment on posts. Teachers can respond to comments.
9. **User Logout**: Secure logout functionality was implemented, clearing tokens and sessions.
10. **Security**: Token and refresh token-based authentication were implemented for enhanced security. Sessions are securely managed, and user data is protected.

**4. Integration and Testing**

In the Integration and Testing phase, all modules and components were integrated to form a complete system. Comprehensive testing was conducted to ensure that the system functions as expected.

**Testing Strategies:**

1. **Unit Testing**: Each feature was tested individually to ensure they function correctly.
2. **Integration Testing**: After individual components were implemented, they were tested together to ensure seamless interaction between the frontend and backend.
3. **Security Testing**: Security measures were tested to ensure that user data is protected and that unauthorized access is prevented.
4. **User Acceptance Testing**: The system was tested by end-users (teachers, students, and admins) to ensure it meets their needs and requirements.

**5. Deployment**

The Deployment phase involved moving the system from the development environment to a live production environment. The system was deployed on a web server, making it accessible to users.

**Deployment Steps:**

1. **Server Setup**: The server environment was configured, including the installation of necessary software like Node.js, databases, and web servers the used one is Vercal.
2. **Code Deployment**: The codebase was uploaded to the server, and all necessary configurations were applied I can’t upload backend code because I’m using some packs that not compatible with Vercal like (Bcypt).
3. **Database Migration**: The database was migrated to the live environment, ensuring all data structures were in place (MongoDB).

**6. Maintenance**

In the Maintenance phase, the system is monitored for any issues or bugs that might arise after deployment. Regular updates and patches are applied to keep the system running smoothly.

**Maintenance Activities:**

1. **Bug Fixes**: Any bugs or issues reported by users are addressed promptly.
2. **Performance Optimization**: The system is continuously monitored for performance, and optimizations are made to improve speed and efficiency.
3. **Security Updates**: Security patches are applied to protect against new vulnerabilities and threats.
4. **User Support**: Ongoing support is provided to users to assist with any issues they may encounter while using the system.

**7. Evaluation**

The Evaluation phase involves assessing the overall performance of Abram's School Management System and determining whether it meets the initial requirements and objectives. Feedback from users is collected and analyzed to identify areas for improvement.

**Evaluation Criteria:**

1. **User Satisfaction**: Surveys and feedback forms are used to gauge user satisfaction with the system's features and usability.
2. **System Performance**: The system's performance is evaluated based on its speed, reliability, and scalability.
3. **Security**: The effectiveness of security measures is reviewed to ensure that user data remains safe.
4. **Feature Effectiveness**: The usefulness and effectiveness of each feature are evaluated, and any necessary enhancements are identified.

**Conclusion**

The Abram's School Management System is a robust and secure platform designed to streamline school management tasks. By following the Waterfall Model, we ensured that each phase of development was completed thoroughly before moving on to the next, resulting in a well-structured and efficient system. The system's features, from user management to classroom activities, are designed to enhance the educational experience for students, teachers, and administrators alike.