Implementation Plan

- [ ] 1. Set up Spring Boot project structure and dependencies
    - Create Spring Boot project with Maven/Gradle build configuration
    - Add dependencies for Spring Security, Spring Data JPA, MariaDB connector, BCrypt
    - Configure application.properties for MariaDB connection and security settings
    - Set up testing framework with JUnit 5 and jqwik for property-based testing
    - Add Micrometer and Actuator dependencies for performance metrics
    - *Requirements: 1.1, 2.1, 4.1, 5.1*
- [ ] 2. Create MariaDB database schema and JPA entities
    - [ ] 2.1 Set up MariaDB database and connection configuration
        - Create database schema with proper indexing for performance
        - Configure HikariCP connection pool for optimal performance
        - Add database migration scripts using Flyway
        - *Requirements: 1.1, 2.1, 4.1*
    - [ ] 2.2 Create User JPA entity with validation
        - Implement User entity with JPA annotations and Bean Validation
        - Add email format validation and password strength checking
        - Configure database constraints and indexes
        - *Requirements: 1.1, 1.3, 1.4*
    - [ ]* 2.3 Write property test for input validation
        - **Property 3: Input validation rejection**
        - **Validates: Requirements 1.3, 1.4**
    - [ ] 2.4 Create Session JPA entity with lifecycle management
        - Implement Session entity with automatic expiration handling
        - Add JPA lifecycle callbacks for session management
        - Configure session cleanup scheduled tasks
        - *Requirements: 2.1, 6.1, 6.2*
    - [ ]* 2.5 Write property test for session lifecycle
        - **Property 18: Session lifecycle management**
        - **Validates: Requirements 6.1, 6.2**
    - [ ] 2.6 Create PasswordResetToken JPA entity
        - Implement token entity with expiration and single-use enforcement
        - Add database constraints for token uniqueness
        - Configure automatic token cleanup
        - *Requirements: 4.1, 4.4, 4.5*

  - [ ]* 2.7 Write property test for reset token behavior
    - **Property 14: Reset token single-use enforcement**
    - **Validates: Requirements 4.5**
- ☐ 3. Implement Spring Security configuration and services
  - ☐ 3.1 Configure Spring Security with custom authentication
    - Set up SecurityConfig with BCryptPasswordEncoder
    - Configure authentication providers and security filters
    - Add CORS and CSRF protection configuration
    - *Requirements: 5.1, 5.2*
  - [ ]* 3.2 Write property test for password hashing
    - **Property 15: Password hashing security**
    - **Validates: Requirements 5.1**
  - ☐ 3.3 Create JWT token service for session management
    - Implement JWT token generation and validation
    - Add token refresh and blacklist functionality
    - Configure token expiration and security settings
    - *Requirements: 4.1, 5.3, 6.1*
  - [ ]* 3.4 Write property test for token randomness
    - **Property 16: Reset token randomness**
    - **Validates: Requirements 5.3**
  - ☐ 3.5 Implement rate limiting with Redis/In-memory cache
    - Create rate limiting interceptor using Spring AOP
    - Add configurable limits with sliding window algorithm
    - Integrate with Spring Cache for performance
    - *Requirements: 5.4*
  - [ ]* 3.6 Write property test for rate limiting
    - **Property 17: Rate limiting enforcement**
    - **Validates: Requirements 5.4**
- ☐ 4. Set up performance monitoring and metrics
  - ☐ 4.1 Configure Micrometer and Actuator endpoints
    - Set up Micrometer with Prometheus registry for metrics collection
    - Configure Actuator endpoints for health checks and monitoring
    - Add custom metrics for authentication operations
    - *Requirements: All requirements (performance monitoring)*
  - ☐ 4.2 Implement performance logging and monitoring

        - Add method-level performance monitoring using @Timed annotations
        - Configure database connection pool metrics
        - Set up authentication success/failure rate metrics
        - *Requirements: 2.1, 2.2, 5.5*
    - [ ] 4.3 Create performance test benchmarks
        - Implement JMH benchmarks for critical authentication paths
        - Add load testing scenarios for concurrent authentication
        - Configure performance thresholds and alerts
        - *Requirements: 2.1, 2.5, 5.4*
- [ ] 5. Checkpoint - Ensure all tests pass
    - Ensure all tests pass, ask the user if questions arise.
- [ ] 6. Implement Spring Data JPA repositories
    - [ ] 6.1 Create UserRepository with custom queries
        - Implement JpaRepository with custom query methods
        - Add optimized queries for email lookup and login timestamp updates
        - Configure query performance monitoring and caching
        - *Requirements: 1.1, 1.2, 2.4*
    - [ ]* 6.2 Write property test for user registration
        - **Property 1: User registration with valid credentials**
        - **Validates: Requirements 1.1**
    - [ ]* 6.3 Write property test for duplicate prevention
        - **Property 2: Duplicate email prevention**
        - **Validates: Requirements 1.2**
    - [ ] 6.4 Create SessionRepository with cleanup scheduling
        - Implement repository with automatic session cleanup
        - Add batch operations for session management
        - Configure database indexes for session queries
        - *Requirements: 2.1, 3.1, 6.1, 6.2*
    - [ ] 6.5 Create AuditLogRepository for security events
        - Implement repository for security event logging
        - Add efficient querying and archival capabilities
        - Configure async logging for performance
        - *Requirements: 3.4, 5.5*
    - [ ]* 6.6 Write property test for audit logging
        - **Property 20: Security audit logging**

        - **Validates: Requirements 3.4, 5.5**

- [ ] 7. Implement authentication service layer
    - [ ] 7.1 Create UserService with registration functionality
        - Implement @Service class with @Transactional methods
        - Add email validation, duplicate checking, and password encoding
        - Integrate with email service for verification
        - *Requirements: 1.1, 1.2, 1.3, 1.4, 1.5*
    - [ ]* 7.2 Write property test for registration email
        - **Property 4: Registration email notification**
        - **Validates: Requirements 1.5**
    - [ ] 7.3 Create AuthenticationService with login functionality
        - Implement Spring Security authentication with custom logic
        - Add failed attempt tracking and account locking mechanisms
        - Integrate with JWT token generation and metrics
        - *Requirements: 2.1, 2.2, 2.3, 2.4, 2.5*
    - [ ]* 7.4 Write property test for valid login
        - **Property 5: Valid login creates session**
        - **Validates: Requirements 2.1**
    - [ ]* 7.5 Write property test for login rejection
        - **Property 6: Invalid authentication rejection**
        - **Validates: Requirements 2.2, 2.3**
    - [ ]* 7.6 Write property test for login timestamps
        - **Property 7: Login timestamp recording**
        - **Validates: Requirements 2.4**
    - [ ]* 7.7 Write property test for account locking
        - **Property 8: Account locking after failed attempts**
        - **Validates: Requirements 2.5**
    - [ ] 7.8 Create SessionService with logout functionality
        - Implement session management with JWT blacklisting
        - Add audit logging and performance metrics
        - Configure async session cleanup
        - *Requirements: 3.1, 3.3, 3.4*
    - [ ]* 7.9 Write property test for session invalidation
        - **Property 9: Session invalidation on logout**

        - **Validates: Requirements 3.1**
    - [ ]* 7.10 Write property test for post-logout access
        - **Property 10: Access denial after logout**
        - **Validates: Requirements 3.3**
- [ ] 8. Implement password reset functionality
    - [ ] 8.1 Create PasswordResetService with request handling
        - Implement @Service with secure token generation
        - Add silent handling for unregistered emails with consistent timing
        - Integrate with email service and rate limiting
        - *Requirements: 4.1, 4.2*
    - [ ]* 8.2 Write property test for reset token generation
        - **Property 11: Password reset token generation**
        - **Validates: Requirements 4.1**
    - [ ]* 8.3 Write property test for silent email handling
        - **Property 12: Silent handling of unregistered emails**
        - **Validates: Requirements 4.2**
    - [ ] 8.4 Create password reset completion functionality
        - Implement token validation and password update
        - Add @Transactional token invalidation after successful reset
        - Configure performance monitoring for reset operations
        - *Requirements: 4.4, 4.5*
    - [ ]* 8.5 Write property test for password reset completion
        - **Property 13: Password reset completion**
        - **Validates: Requirements 4.4**
- [ ] 9. Implement session management and security features
    - [ ] 9.1 Create JWT authentication filter
        - Implement OncePerRequestFilter for JWT validation
        - Add automatic session extension and blacklist checking
        - Configure filter chain with performance monitoring
        - *Requirements: 6.1, 6.2*
    - [ ] 9.2 Create DeviceDetectionService with notification
        - Implement device fingerprinting and detection logic
        - Add async security notification email sending
        - Configure device tracking with privacy considerations
        - *Requirements: 6.5*

  - [ ]* 9.3 Write property test for device notifications
    - **Property 19: New device notification**
    - **Validates: Requirements 6.5**

- [ ] 10. Create REST controllers with Spring Boot
  - [ ] 10.1 Implement AuthController with registration endpoint
    - Create @RestController with @PostMapping for /api/auth/register
    - Add @Valid request body validation and proper error handling
    - Configure response DTOs and HTTP status codes
    - *Requirements: 1.1, 1.2, 1.3, 1.4, 1.5*
  - [ ] 10.2 Implement login endpoint with security
    - Create POST /api/auth/login with @RateLimited annotation
    - Add JWT token generation and security headers
    - Configure authentication success/failure metrics
    - *Requirements: 2.1, 2.2, 2.3, 2.4, 2.5*
  - [ ] 10.3 Implement logout endpoint with cleanup
    - Create POST /api/auth/logout with JWT blacklisting
    - Add proper response handling and audit logging
    - Configure async session cleanup
    - *Requirements: 3.1, 3.3, 3.4*
  - [ ] 10.4 Implement password reset endpoints
    - Create POST /api/auth/reset-request and /api/auth/reset-confirm
    - Add comprehensive validation and security measures
    - Configure rate limiting and performance monitoring
    - *Requirements: 4.1, 4.2, 4.4, 4.5*
  - [ ] 10.5 Implement session validation endpoint
    - Create GET /api/auth/session with JWT validation
    - Add session extension and health check logic
    - Configure caching for performance optimization
    - *Requirements: 6.1, 6.2*
- [ ] 11. Create Thymeleaf templates and frontend integration
  - [ ] 11.1 Create registration page with Thymeleaf
    - Implement registration form template with server-side validation
    - Add Bootstrap styling and client-side validation
    - Configure CSRF protection and error display
    - *Requirements: 1.1, 1.3, 1.4*
  - [ ] 11.2 Create login page with security features

        - Implement login form template with credential input
        - Add "remember me" functionality and error handling
        - Configure rate limiting display and account lockout messages
        - *Requirements: 2.1, 2.2, 6.3*

    - [ ] 11.3 Create password reset page templates

        - Implement reset request and confirmation form templates
        - Add proper validation feedback and user guidance
        - Configure secure token handling in URLs
        - *Requirements: 4.1, 4.3, 4.4*

    - [ ] 11.4 Create authentication state management

        - Implement JavaScript for JWT token handling
        - Add automatic logout on token expiration
        - Configure axios interceptors for API authentication
        - *Requirements: 3.3, 6.2*

- [ ] 12. Implement Spring Boot email service integration

    - [ ] 12.1 Configure JavaMailSender with email templates

        - Set up Spring Boot mail configuration with SMTP
        - Create Thymeleaf email templates for verification
        - Add async email sending with @Async annotation
        - *Requirements: 1.5*

    - [ ] 12.2 Create password reset email service

        - Implement email service with secure reset links
        - Add HTML email templates with proper styling
        - Configure email delivery monitoring and retry logic
        - *Requirements: 4.1*

    - [ ] 12.3 Create security notification email service

        - Implement new device login notification system
        - Add security alert templates with device information
        - Configure email rate limiting and delivery tracking
        - *Requirements: 6.5*

- [ ] 13. Add comprehensive error handling and security

    - [ ] 13.1 Implement @ControllerAdvice global exception handler

        - Create centralized exception handling for all controllers
        - Add proper error logging with structured logging (Logback)
        - Configure user-friendly error responses and security headers
        - *Requirements: All requirements*

    - [ ] 13.2 Add input sanitization and validation

        - Implement comprehensive Bean Validation across all DTOs
        - Add XSS prevention with OWASP Java Encoder
        - Configure SQL injection prevention with JPA parameterized queries
        - *Requirements: 1.3, 1.4, 5.4*
    - [ ] 13.3 Configure production security settings
        - Set up HTTPS configuration and security headers
        - Configure database connection encryption
        - Add application security monitoring and alerting
        - *Requirements: 5.1, 5.2, 5.4, 5.5*
- [ ] 14. Performance optimization and monitoring setup
    - [ ] 14.1 Configure database performance optimization
        - Set up connection pool tuning and query optimization
        - Add database query performance monitoring
        - Configure read replicas for scaling (if needed)
        - *Requirements: All requirements (performance)*
    - [ ] 14.2 Set up application performance monitoring
        - Configure APM with Micrometer and Prometheus
        - Add custom business metrics dashboards
        - Set up alerting for performance thresholds
        - *Requirements: All requirements (monitoring)*
- [ ] 15. Final checkpoint - Ensure all tests pass
    - Ensure all tests pass, ask the user if questions arise.