1. PURPOSE

This procedure defines the processes and rules to be followed for software to be developed and/or produced by the Information Technologies Department in order to meet software/program requests submitted by academic and administrative staff, students, prospective students, or external organizations, as well as to address identified deficiencies within the available resources of Cyprus Aydın University.

2. SCOPE

This procedure applies to all software developed and/or produced by the Information Technologies Department.

3. IMPLEMENTATION

3.1 Receiving / Identifying and Evaluating Software Requests

3.1.1 Software development needs of students and academic/administrative staff, as well as changes arising from regulations, directives, and official correspondence of the institutions and organizations for which the University is responsible, are evaluated by the relevant units of the Information Technologies Department and senior management. Necessary modifications and updates are determined, a schedule is prepared, and records are monitored through the UBIS Timeline System.

3.1.2 The Information Technologies Department may decide on updates required due to developments in software and security technologies or in order to improve user-friendliness. A schedule is then prepared for the implementation of such updates.

3.2 Fulfillment of Software Requirements

3.2.1 Software shall be developed in compliance with the requirements specified in the Secure Coding Principles.

3.2.2 For the software that has been approved and scheduled, a relevant process shall be created in the Timeline application within the UBIS system and assigned to the developer(s) responsible for its implementation.

3.2.3 Application development activities shall be carried out in an environment separate from the production environment. Software developers may access this development environment only through the **SVN method** and shall develop applications using **security-tested frameworks** within areas specifically authorized for them.

3.2.4 After the application has been developed, it shall be tested by another software developer assigned to perform testing activities.

3.2.5 If any issues are identified, the errors and problems shall be documented and communicated to the software developer responsible for the application, and the process shall continue from the previous step.

3.2.6 If no issues are identified, testing authorization shall be granted to selected personnel from

the unit that will use the application, and the application shall be tested by those personnel.

3.2.7 If any issues are identified during user testing, the errors shall be documented and communicated to the software developer responsible for the application, and the process shall continue from the relevant step.

3.2.8 If no issues are identified, the software shall be deployed, and the relevant process shall be closed in the Timeline application upon completion of the software development activities.

3.3 Secure Coding Principles

3.3.1 The software development environment and the database environment shall be maintained as separate and independent environments. Critical information related to these environments (such as server names, server versions, software versions, etc.) shall be concealed.

3.3.2 The software development environment shall be defined within a separate virtual host environment in such a way that access other than through the SVN method is restricted to software developers. Access shall be controlled through physical firewalls and network switches.

3.3.3 The database environment shall be managed through firewalls and network switches in a manner that permits access only to database administrators.

3.3.4 Code shall be transferred to the software development environment exclusively through the SVN method. This method ensures version control of the software repository.

3.3.5 A separate username, password, and authorization profile shall be assigned to each software developer.

3.3.6 User inputs within the application shall be coded with due consideration to input sanitization, filtering mechanisms, and SQL Injection prevention controls.

3.3.7 All access requests to the application shall be subject to authorization controls both at the request stage and at the response stage.

3.3.8 Every request made by a user within the application environment shall be recorded in audit logs and processed only after authorization checks have been performed.

3.3.9 Security scans of all software components, including the application framework, database, application server, and web server, shall be maintained at the highest security level.

3.3.10 All failed operations occurring during the use of the application shall be logged and reported to the application developers.

3.3.11 Default user accounts shall be removed from the system, database, and application.

3.3.12 To prevent web pages containing sensitive information from being cached by browsers, appropriate HTTP/HTML headers such as autocomplete, cache-control, and pragma shall be used.

3.3.13 Strong encryption algorithms shall be used to ensure secure web traffic (SSL).

3.3.14 Passwords entered by users shall be masked in all password fields and shall not be displayed in plain text.

3.3.15 The use of weak passwords shall not be permitted.

3.3.16 The HTTPS protocol shall be used within the application.

3.3.17 Server-side authentication shall be implemented for access to all non-public resources and pages.

3.3.18 User passwords shall be stored in encrypted form. Salt values shall also be used when generating password hash data.

3.3.19 Users shall be required to change their passwords upon first use of the credentials assigned to them. Passwords shall contain at least one uppercase letter, one lowercase letter, one numeric digit, and one special character, and shall be a minimum of eight (8) characters in length.

3.3.20 Accounts shall be locked after a specified number of unsuccessful login attempts. When an incorrect username or password is entered, a generic error message shall be displayed in order to conceal the source of the error and prevent social engineering attacks.

3.3.21 A password validity period shall be defined. This period shall be 90 days.

3.3.22 The current password shall be required when changing a password.

3.3.23 For each user session, unique time-limited cookies and sessions shall be created. Upon logout, all cookies and sessions associated with that session shall be terminated.

3.3.24 Role-based authorization shall be implemented for access control.

3.3.25 User permissions shall be assigned only by system administrators or duly authorized personnel.

3.3.26 In the event of an attack, security measures such as CAPTCHA and similar protection mechanisms shall be activated.