**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Identify Design Elements**

**for**

**The Automatic**

**Attendance Checking System**

**Version 1.1**

**Prepared by Huynh Vinh Nam**

**Le Huy Duc**

**Cao Phuong Linh**

**OOAD Group 2**

**15-Dec-2018**

**Table of Contents**

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Reason for Changes** | **Version** |
| Huynh Vinh Nam | 15-Dec-2018 | Create document template | 1.0 |
| Huynh Vinh Nam | 15-Dec-2018 | Add Picture from model file | 1.1 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**1. Introduction**

**1.1. Purpose**

This is a report on the subject Object-oriented Analysis and Design of group two, class ICT-BI7 about System Architectural Analysis of the project AACS.

The report is written based on the reporting format “IEEE Std 830-1998 IEEE Recommended Practice for Software Requirements Specifications”. Content-based report is accepted and be satisfied with group meeting minute on 13-Dec-2018 (Viber teleconference).

This document is used to analyze interactions of analysis classes to identify design model elements.

**1.2. Intended Audience and Reading Suggestions**

**Role Software Architect:** The software architect role leads and coordinates technical activities and artifacts throughout the project. The software architect establishes the overall structure for each architectural view: the decomposition of the view, the grouping of elements, and the interfaces between these major groupings. Therefore, in contrast to the other roles, the software architect's view is one of breadth as opposed to one of depth.

*The different types of reader that the document is intended for are:*

**● Project managers:** who manage and take respond for the quality of the system. Project

managers should read the whole document for planning and assigning work.

**● Developers:** Dev is the person who implement the system from the design and documents into a runnable version. Dev have to read the whole document to implement the right system.

**● Documentation writers:** who will write the future document (report, minutes).

Documentation writers should read to understand the Use Case Main Diagram part.

*The content of report includes five main parts:*

**● Part 1 Subsystem Context Diagrams:**

**● Part 2 Analysis Class to Design Element Map**

**● Part 3 Design Element to Owning Package Map**

**● Part 4 Architectural Layers and Their Dependencies:**

**● Part 5 Packages and Their Dependencies:**

**1.3. Product Scope**

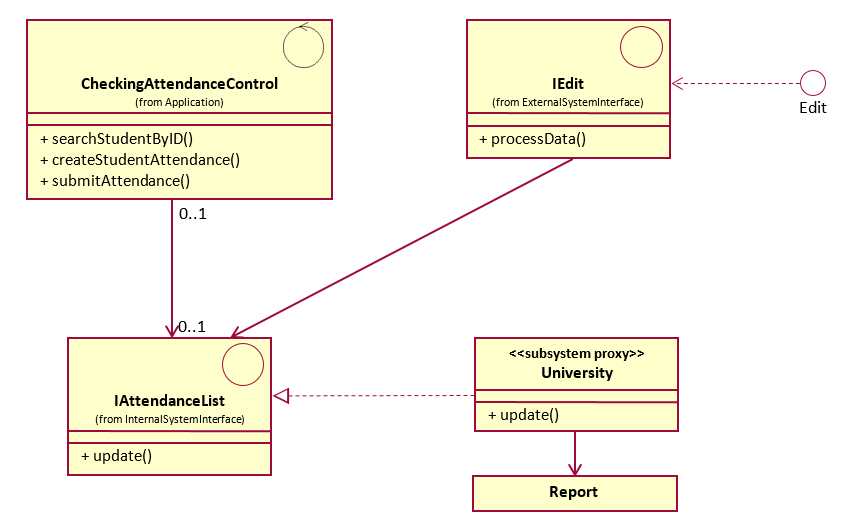
The software’s main users are students and lecturers. Software will create an environment where user (student) can check for the attendance and user (lecturer) can view and/or manage the attendance list in the course(s).

**1.4. References**

[1] Form of presentation IEEE. IEEE Std 830-1998 IEEE Recommended Practice for Software Requirements Specifications. IEEE Computer Society, 1998.

**2. Subsystem Context Diagrams**

**2.1. University Subsystem**



*Figure 1: University Subsystem diagram*

**3. Analysis Class to Design Element Map**

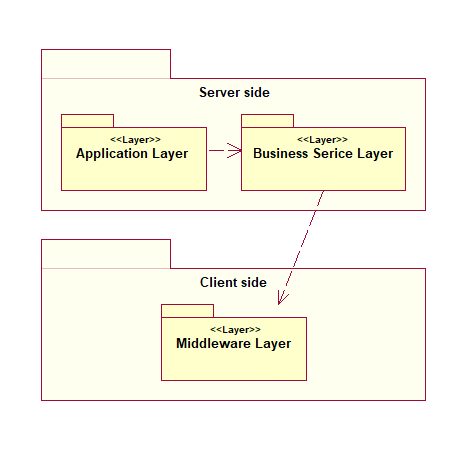
|  |  |
| --- | --- |
| **Analysis Class** | **Design Element** |
| GenerateQRCodeControl | SelectCourseControl |
|  | SelectCreateNewAttendanceControl |
|  | RequireConfirmationControl |
| CheckingAttendanceControl | SearchStudentByIDControl |
|  | CreateStudentAttendanceControl |
|  | SubmitAttendanceControl |
| ViewControl | SelectManageAttendanceControl |
|  | SelectViewControl |
| EditControl | SelectEditControl |
|  | ProcessDataControl |
|  | UpdateHistoryOfChangeControl |
|  | UpdateListControl |
|  | LoadOlderVersionHistoryOfChangeControl |
| GenerateQRCodeBoundary | CreateNewAttendanceForm |
|  | ManageAttendanceForm |
| ForgotPasswordBoundary | ForgotPasswordForm |
| SignInBoundary | SignInForm |
| LecturerProfile | LecturerProfile |
| DB | (Finish with Database design) |
| Profile | (Finish with Database design) |
| UniversityDB | (Finish with Database design) |

**4 Design Element to Owning Package Map**

|  |  |
| --- | --- |
| **Design Element** | **Owning Package** |
| SelectCourseControl | Application |
| SelectCreateNewAttendanceControl | Application |
| SelectManageAttendanceControl | Application |
| SelectViewControl | Application |
| SelectEditControl | Application |
| RequireConfirmationControl | Application |
| SearchStudentByIDControl | Application |
| CreateStudentAttendanceControl | Application |
| SubmitAttendanceControl | Application |
| SelectManageAttendanceControl | Application |
| LecturerProfile | Application |
| GenerateQRCodeForm | Middleware:: Presentation |
| CreateNewAttendanceForm | Middleware:: Presentation |
| CheckingAttendanceForm | Middleware:: Presentation |
| ManageAttendanceForm | Middleware:: Presentation |
| ViewForm | Middleware:: Presentation |
| EditForm | Middleware:: Presentation |
| ForgotPasswordForm | Middleware:: Presentation |
| SignInForm | Middleware:: Presentation |
| LecturerProfile | Business Service::Domain::Lecture |
| CourseCatalogue | Business Service::Domain::Lecture |
| StudentProfile | Business Service::Domain::University |
| Report | Business Service::Domain::University |
| DB | Business Service::Persistence |
| Profile | Business Service::Persistence |
| UniversityDB | Business Service::Persistence |
| DatabaseManagerSubsystem | Business Service |
| IDatabaseManagerInterface | Business Service::ExternalSystemInterfaces |
| SecurityManagerSubsystem | Business Service::Security |
| ISecurityManagerInterface | Business Service::Security::SecurityInterface |

**5. Architectural Layers and Their Dependencies**

**5.1. Layer Dependencies Diagram**

****

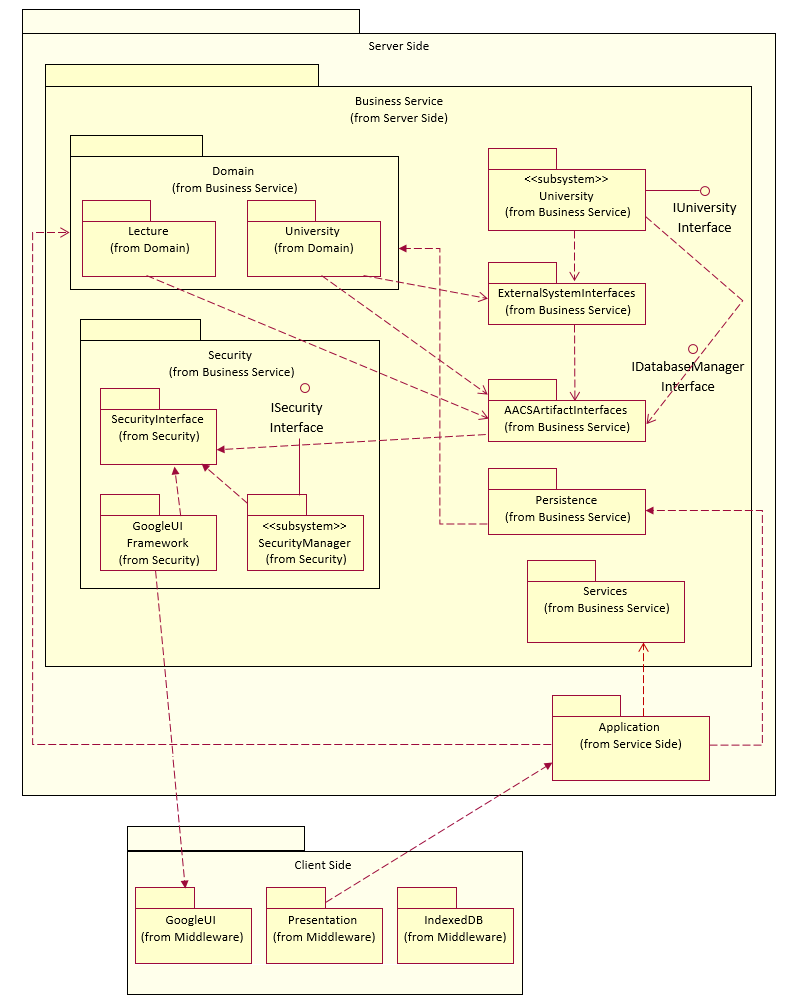
*Figure 2: Layer Dependencies Diagram*

**5.2. Layer Descriptions**

* **Server Side:** Server layer supports several different application servers, where “application” includes static web pages. The existence of a server is known to the network server. In general servers are managed by the network server, but application programs may partly take over this responsibility.
* **Client Side:** The client layer is where the user accesses the application. The server layer accepts requests through the internet connection from client layer and passes these requests to the appropriate agent. The server then relays the response from the agent back to the client layer. In this cases, the client is simply the QR scanning application (to display QR code) and the browser (to check attendance by access the generated URL and then display the server response) on Android devices.
* **Application:** The Application layer contains application-specific design elements.
* **Business Services:** The Business Services layer contains business-specific elements that are used in serveral applications.
* **Middleware:** Provides utilities and platform-independent services.

**6. Packages and Their Dependencies**

**6.1. Package Dependencies Diagram**

****

*Figure 3: Package Dependencies Diagram*

**6.2. Package Descriptions**

This diagram describes how packages on a same layer depend on each other and depend on other packages from other layers. A dependency exists between two packages if changes to one package may cause changes to the other. It also show how visibility can be defined.

* **Server Side:** Server layer supports several different application servers, where “application” includes static web pages.
* **Business Service:** The Business Services layer contains business-specific elements that are used in several applications.
* **Domain:** Contains packages containing the design elements to support Lecture, University and their management.
* **Security:** Contains design elements that implement the security mechanism.
* **SecurityInterface:** Contains the interfaces that provide clients access to security services.
* **SecurtiyManager Subsytem:** Provides the implementation for the core security services.
* **ISecurityInterface:** Defines a set of behaviors offered by SecurtiyManage subsytem.
* **GoogleUI Framework:** This package comprises a whole framework for user interface management.
* **University subsystem:** Encapsulates communication with all external bank systems.
* **ExternalSystemInterface:** The external system access classes were partitioned into this package.
* **AACSArtifactInterfaces:** This package contains the core AACS abstractions, which implemented as interfaces.
* **Persistence:** Contains the design elements to persist specific objects within the system.
* **Service:** Contains design classes to provide system-level classes for maintenance purposes.
* **Application:** This package contains design classes for major processing functionality within the system.
* **Client Side:** The client layer is where the user accesses the application.
* **GoogleUI:** The package contains some basic design elements of Google JSON.
* **Presentation:** Contains the design elements, which are mapped from Boundary classes, for each of the forms that the users use to communicate with the System.
* **IndexedDB:** This package contains design elements to support Offline Database, which is available for Hybrid app.