KINGDOM OF SAUDI ARABIA
Ministry of Higher Education
Al-Imam Mohammad University
College of Computer & Information Sciences

المملكة العربية السعودية
وزارة التعليم العالي
جامعة الإمام محمد بن سعود
الإسلامية
كلية علوم الحاسب والمعلومات

Second term 1441/2020

Software Engineering (CS- 310)
BSCS- Section: 171

Project-Phase No: 02
SATS
(DD)
Submitted By

1-Hashem Ali Bajabaa (437015783) Coordinator
2-Abdulilah Saleh Alqasem (439014750)
3-Saleh Mohammad Sawab (436020675)
4-Abdulaziz Aldukhaiel (439012555)
5-Mohammad Abdulaziz Zouman (439012798)

Supervisor
Dr. Sultan Alqahtani

Table of Contents

Abstract

This is a design document (DD) that will describe the SATS project in details of design architecture, design model, class hierarchy and how the sub-system will communicate with each other as a whole system. It will include three chapters introduction, design include system architecture and system design and last chapter will be conclusion. This document will give a good understanding of how the system will be like.

Introduction

This section gives a scope description and overview of everything included in this DD document. Also, the purpose for this document is described and a list of abbreviations and definitions is provided.

Purpose

The purpose of this DD document is to give a detailed description of the system high level and medium level design, detailed design for the "Students Attendance Tracking System" (SATS) software. It will illustrate the purpose and complete declaration for the development of system. Also, it will explain what the system should do and how it will do it. This document is meant to be proposed to a customer for its approval and a reference for the developing team to build the system.

Scope

This project will improve the students attendance taking process in universities. Currently, attendance tracking methods in universities are time-consuming which waste valuable lecture time. The project main stakeholders are University Professors and students as they'll benefit tremendously from such software. It will keep professors and students focused on learning and not waste their time doing routine nonproductive task such as taking attendance. Instead of calling each student name in every class, which sometimes can be challenging for foreign professors pronouncing some difficult names, attendance can simply be taken by students scanning badges upon entry. Another benefit this software will produce is the accuracy and fairness of attendance records as there is no room for human error. This software will give immediate feedback to students if they are in the wrong classroom upon scanning.

The main goal for SATS is to build a system that can track attendance effortlessly, organized convenient, fairly and improves the process of attendance tracking and monitoring in Imam Mohammad bin Saud Islamic University. SATS is going to have multiple objectives. As a system it will have to create a database with functions to read, write, and manipulate the data. Using Excel file to record students attendance and generate report for professors to view. Dealing with scan code reader to scan students IDs and setup the device.

Definitions, Acronyms, and Abbreviations

Term	*Definition*
User	*Someone who interacts with the mobile phone application*
Stakeholder	*Any person who has interaction with the system who is not a developer*
MVC	*Model view controller*
DD	*Document design*
SATS	*Students Attendance Tracking System*

References

[1] IEEE Software Engineering Standards Committee, "IEEE Std 830-1998, IEEE Recommended Practice for Software Requirements Specifications", October 20, 1998.

Overview

This DD document will let the customer and developing team to understand and have a good knowledge about the system design in details. As we going through this document we will approach and elucidation introduction as first chapter. High-level and low-level diagrams, architecture pattern, briefly describing the role of each class as well as the methods included as second chapter. Third chapter will be a conclusion of the document.

System architecture

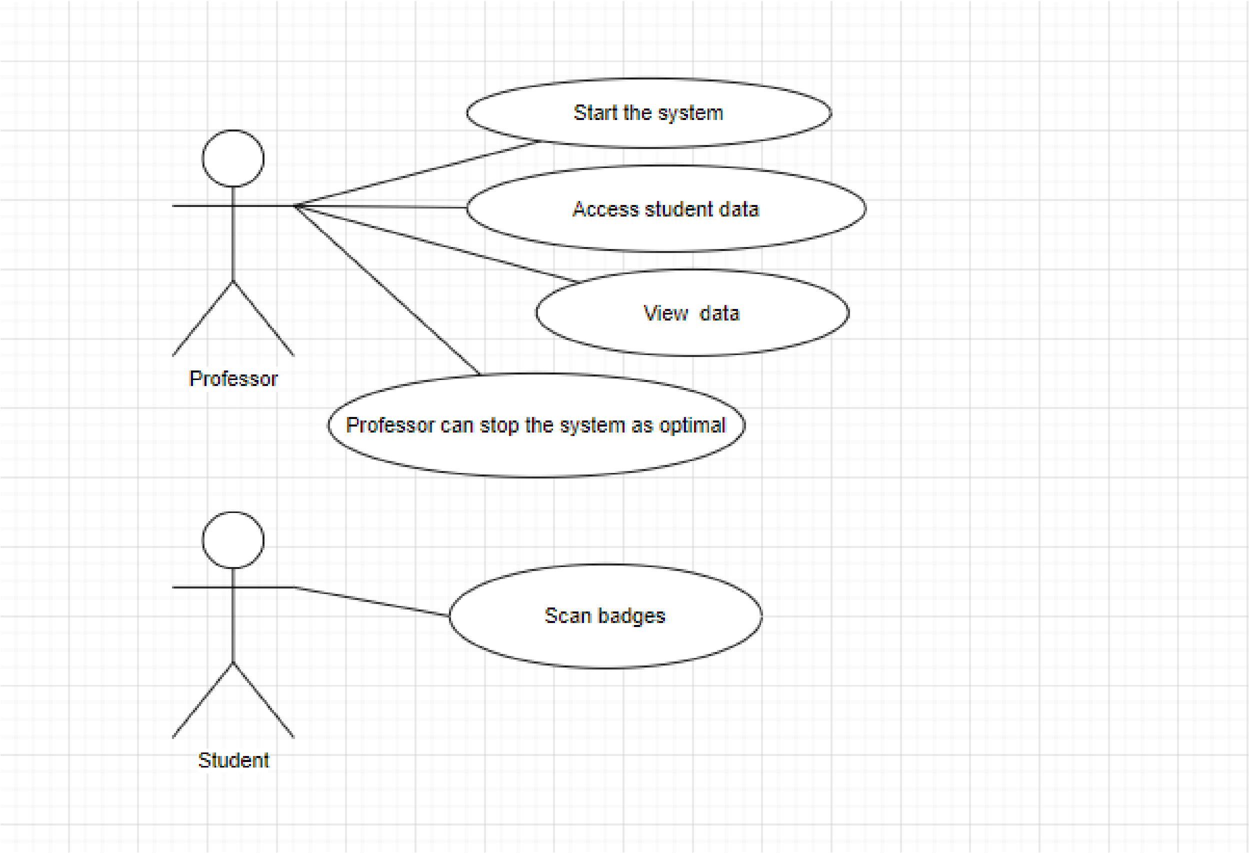
The system architecture that we found fit our project in convenient way is MVC pattern.

We will be divided the classes into three main classes. First module class that will be Responsible to dealing with the database and manipulating data. Second, view class will represent the output in excel file as a report. Third, the whole software classes will be controlled by the controller class that will help to isolate the other classes and make them contact in one class that will improves the work flow and data between the classes**.**

High Level and Low-Level Design

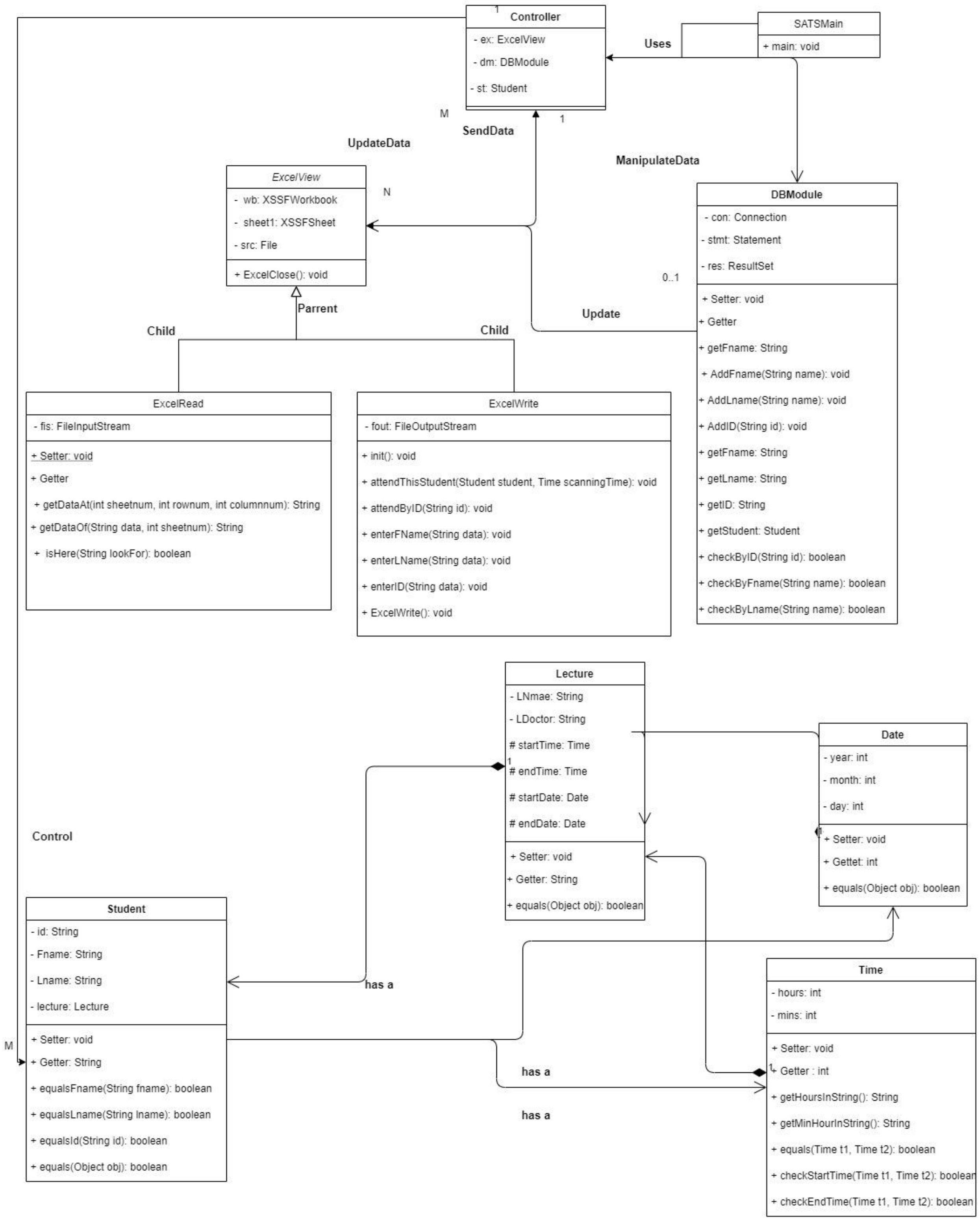
Use-case Diagram

Use case diagram is a representation of user interaction with the system in SATS.



Class Diagram

The class diagram shows the structure of SATS in program units.


Controller
- ex: ExcelView
- dm: DBModule
- st: Student
SATSMain
+ main: void
Uses
1
M
1
SendData
UpdateData
ManipulateData
N
ExcelView
- wb: XSSFWorkbook
- sheet1: XSSFSheet
- src: File
+ ExcelClose(): void
DBModule
- con: Connection
- stmt: Statement
- res: ResultSet
+ Setter: void
+ Getter
+ getFname: String
+ AddFname(String name): void
+ AddLname(String name): void
+ AddID(String id): void
+ getFname: String
+ getLname: String
+ getID: String
+ getStudent: Student
+ checkByID(String id): boolean
+ checkByFname(String name): boolean
+ checkByLname(String name): boolean
0..1
Update
Parrent
Child
Child
ExcelRead
- fis: FileInputStream
+ Setter: void
+ Getter
+ getDataAt(int sheetnum, int rownum, int columnnum): String
+ getDataOf(String data, int sheetnum): String
+ isHere(String lookFor): boolean
ExcelWrite
- fout: FileOutputStream
+ init(): void
+ attendThisStudent(Student student, Time scanningTime): void
+ attendByID(String id): void
+ enterFName(String data): void
+ enterLName(String data): void
+ enterID(String data): void
+ ExcelWrite(): void
Lecture
- LNmae: String
- LDoctor: String
# startTime: Time
# endTime: Time
# startDate: Date
# endDate: Date
+ Setter: void
+ Getter: String
+ equals(Object obj): boolean
1
Date
- year: int
- month: int
- day: int
+ Setter: void
+ Gettet: int
+ equals(Object obj): boolean
Control
Student
- id: String
- Fname: String
- Lname: String
- lecture: Lecture
+ Setter: void
+ Getter: String
+ equalsFname(String fname): boolean
+ equalsLname(String lname): boolean
+ equalsId(String id): boolean
+ equals(Object obj): boolean
M
has a
has a
has a
Time
- hours: int
- mins: int
+ Setter: void
+ Getter : int
1
+ getHoursInString(): String
+ getMinHourInString(): String
+ equals(Time t1, Time t2): boolean
+ checkStartTime(Time t1, Time t2): boolean
+ checkEndTime(Time t1, Time t2): boolean

Class Method Descriptions

In this section, each class method is briefly described with respect to visibility, parameters, and return type.

Controller Class

The Controller class interacts with the DBModule class and the ExcelView class. The Controller class controls all actions performed on the system and passes these interactions to the ExcelView and the DBModule.

DBModule Class

The DBModule class is used to add or check students data. The class DBModule is manipulated by class Controller who has one attribute of type DBModule. The class DBModule has methods to add students name and ID, and methods to check certain name or ID.

Class	DBModule
Method	addFname
Visibility	public
Return type	void
Parameters, types	name : String
Description	This method is used to add a student's first name.

Class	DBModule
Method	addLname
Visibility	public
Return type	void
Parameters, types	name : String
Description	This method is used to add a student's last name.

Class	DBModule
Method	addID
Visibility	public
Return type	void
Parameters, types	id : String
Description	This method is used to add a student’s ID number.

Class	DBModule
Method	checkByID
Visibility	public
Return type	boolean
Parameters, types	id : String
Description	This method is used to check if the ID of student exists.

Class	DBModule
Method	CheckByFname
Visibility	public
Return type	boolean
Parameters, types	name : String
Description	This method is used to check if the first name of student exists.

Class	DBModule
Method	checkByLname
Visibility	public
Return type	boolean
Parameters, types	name : String
Description	This method is used to check if the last name of student exists.

Date Class

The Date class is designed to be used by class Lecture as a composition relation. Class Date has three attributes year, month, day and one method (equals) excluding setters and getters. Class Lecture has two attributes with type Date to get exact dates of when the lecture begins and a date when it will end.

Class	Date
Method	equals
Visibility	public
Return type	boolean
Parameters, types	obj : Object
Description	This method compares the date of scan with the lecture's start date.

Time Class

The Time class is designed to be used by class Lecture as a composition relation. Class Time

Has two attributes hours and mins also it has five methods excluding setters and getters. Class Lecture has two attributes with type Time to get exact time of when the lecture begins and a time for when it ends.

Class	Time
Method	checkStartTime
Visibility	public
Return type	boolean
Parameters, types	t1 : Time, t2 : Time
Description	This method compares the lecture time and the time of scan to determine if scan time is before or after start time.

Class	Time
Method	equals
Visibility	public
Return type	boolean
Parameters, types	t1 : Time, t2 : Time
Description	This method compares the time of scan with the lecture's start time.

Lecture Class

The Lecture class maintains the states and attributes of a lecture, which are the lecture name (course name), the name of doctor of the lecture, start time, end time of the lecture, start date and end date of the lecture. The class Student has an array of type Lecture so every student has its own lectures or courses.

Class	Lecture
Method	equals
Visibility	public
Return type	boolean
Parameters, types	obj : Object
Description	This method checks if a certain lecture is in the database or not.

Student Class

The Student class maintains the states and attributes of a college student, these attributes are first name, last name, ID and an array of lectures based on the courses registered by the student. Every added student is saved on the database and controlled by the Controller class.

Class	Student
Method	equalsFname
Visibility	public
Return type	boolean
Parameters, types	fname : String
Description	This method searches student first name on the database.

Class	Student
Method	equalsLname
Visibility	public
Return type	boolean
Parameters, types	lname : String
Description	This method searches student last name on the database.

Class	Student
Method	equalsId
Visibility	public
Return type	boolean
Parameters, types	id : String
Description	This method searches student ID on the database.

ExcelRead Class

The ExcelRead class its job is to read the information from the excel file, using three methods to do so, method getDataAt, getDataOf and isHere purpose is to read from the excel file each method with different way. The ExcelRead class is a subclass of the ExcelView class.

Class	ExcelRead
Method	getDataAt
Visibility	public
Return type	String
Parameters, types	sheetnum : int, rownum : int, columnnum : int
Description	This method reads the information from the excel file by selecting the sheet number, row number and column number.

Class	ExcelRead
Method	isHere
Visibility	public
Return type	boolean
Parameters, types	lookFor : String
Description	This method checks or searches the excel file for either the first name, last name or the ID.

ExcelWrite Class

The ExcelWrite class lets enter information into the excel file, using five methods you can attend specific student by using method (attendThisStudent), attend by ID using method (attendByID), entering a first name in the excel file using method (enterFName), entering a last name in the excel file using method (enterLName), or entering an ID in the excel file using method (enterID). The ExcelWrite class is a subclass of the ExcelView class.

Class	ExcelWrite
Method	attendThisStudent
Visibility	public
Return type	void
Parameters, types	student : Student, scanningTime : Time
Description	This method attends a specific student with the time of scan.

Class	ExcelWrite
Method	attendByID
Visibility	public
Return type	void
Parameters, types	id : String
Description	This method attends a specific ID with the time of scan.

ExcelView Class

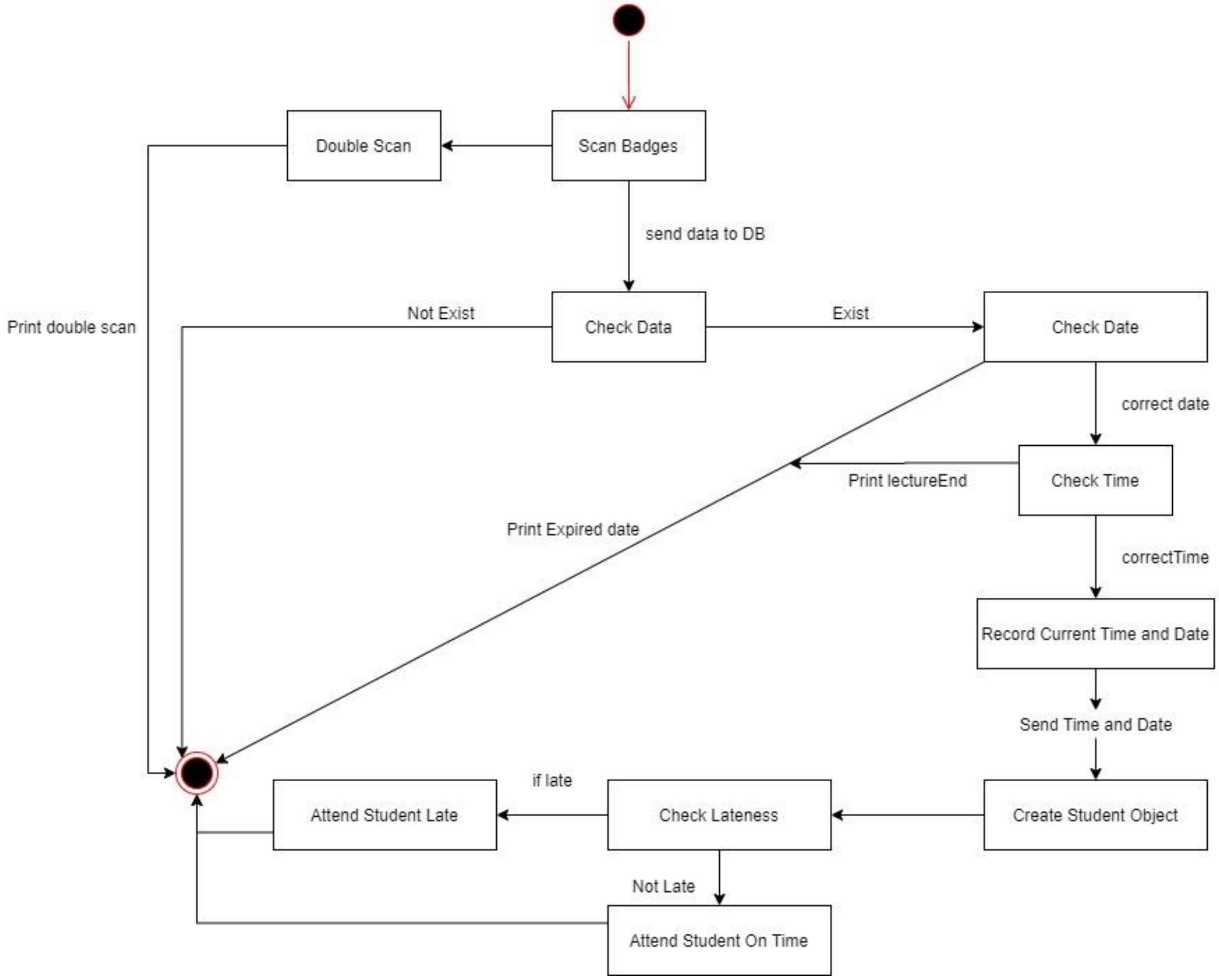
The ExcelView class is the superclass of the two classes ExcelRead and ExcelWrite, the ExcelView class purpose is to just view or display the excel file.

Class	ExcelView
Method	ExcelClose
Visibility	public
Return type	void
Parameters, types	none
Description	This method purpose is to just close the excel file.

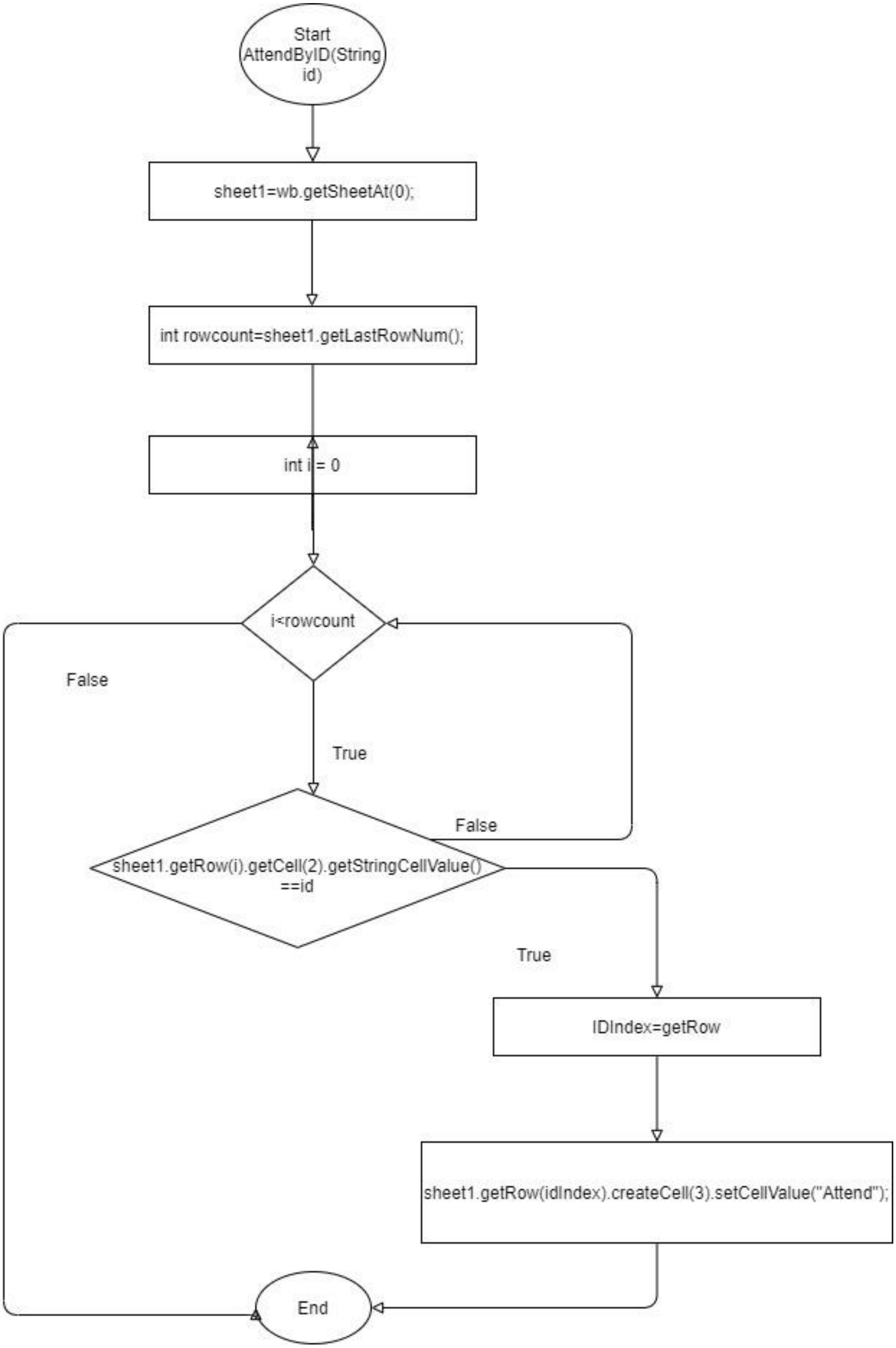
Detailed Design

his section contains diagrams of low-level in details describing SATS.

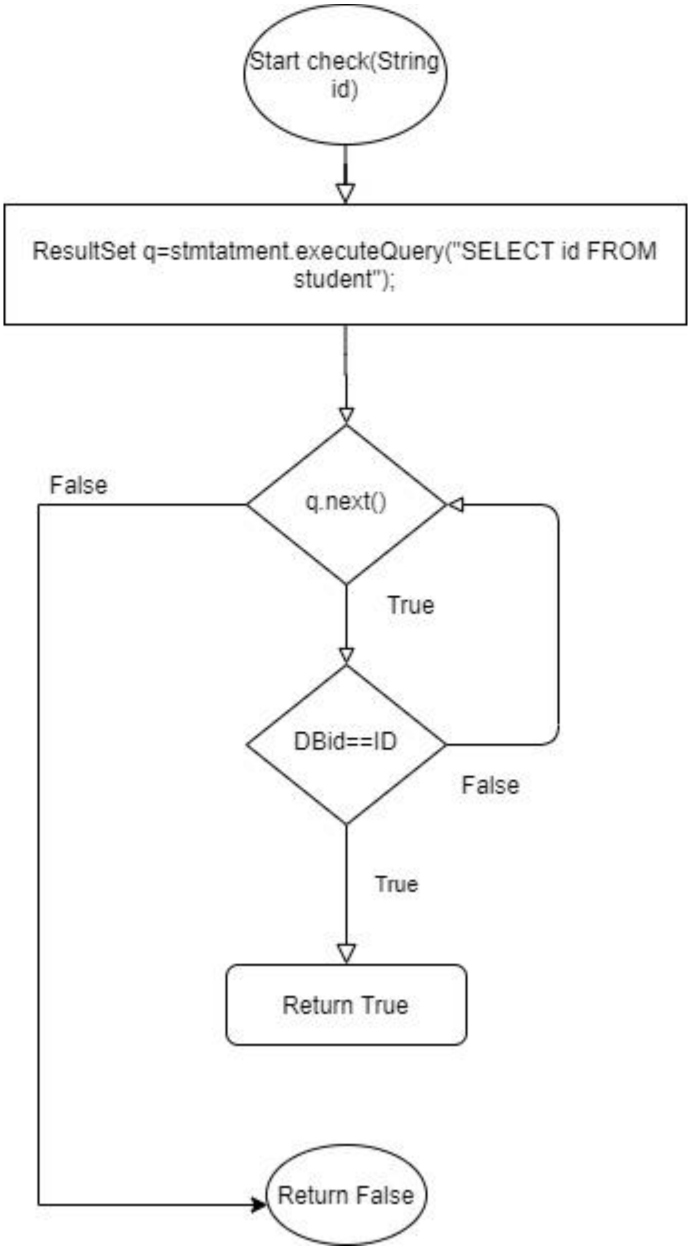
This is an activity chart that explain the activities of the system when start scanning.



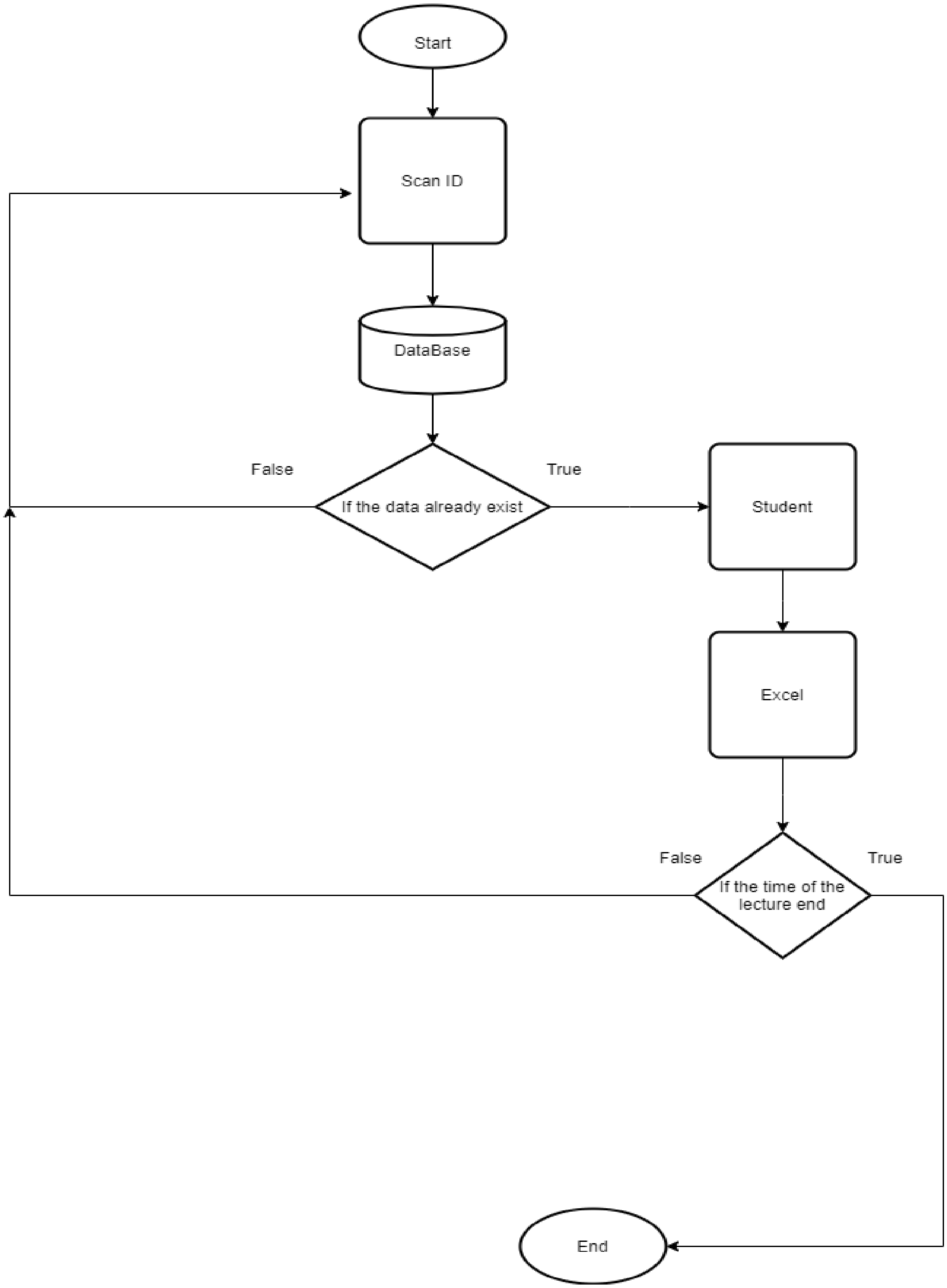
This flow chart explains in details how AttendByID() works.


Start AttendByID(String id)
sheet1=wb.getSheetAt(0);
int rowcount=sheet1.getLastRowNum();
int i = 0
i<rowcount
False
True
sheet1.getRow(i).getCell(2).getStringCellValue() ==id
False
True
IDIndex=getRow
sheet1.getRow(idIndex).createCell(3).setCellValue("Attend");
End

Shows the details of the check(String id) method that manipulate the database**.**



This is a Flowchart of the whole system



Conclusion

In conclusion, our DD will gives a full knowledge of the software in low-level, high-level, class hierarchy and the system architecture. We provide an abstraction, introduction that help to understand the DD and SATS. We also, includes class diagram, use cases, state diagram, sequence diagram, methods and classes description in order to approach the software correctly and as requirement needs.