**ONLINE WATER SUPPLYING SYSTEM**

**ABSTRACT**

Android College Management system is an android application which is helpful for students as well as the colleges. In the existing system all the activities are done manually. It is very costly and time consuming. In our proposed system, students can view attendance using Android phones. The data will be stored in the college server. The faculty can login into their college account through the app itself and update the attendance. In this system, students have easy access for viewing the attendance,home works provided their authentications are correct and they are not permitted to update the attendance, home works and feedback.

The proposed work has two modules: 1.Student 2.Admin. In the student’s module, student need to register their roll no, college registration number, student name. Admin module maintains the student’s attendance details. Other than this the advanced features are: In case of natural calamities such as floods, etc. notification to students will be sent from admin office through app directly. Any new notice for a particular semester will be uploaded by professor through application notifying to respective semester students. Student’s attendance is monitored by the application.

**INTRODUCTION**

The main objective of this system is to computerize the entire operation such as information maintenance and the user ordering process. The system is developed as a mobile app for the user compatibility. The user can use this application and complete their need through online. This online application provides the entire information about the school, staff and student details can also be maintained through online. Through this system the faculty can take the attendance from anywhere through the online process. The project is aimed to develop by **JAVA and XML** as Front end and **MS SQL SERVER** as Back end. The back end is used to store the information in this system.

**SYSTEM SPECIFICATION**

**HARDWARE SPECFICATION:**

* System : Pentium IV 2.4 GHz.
* Hard Disk : 180 GB.
* Floppy Drive : 1.44 Mb.
* Ram : 8 GB.

**SOFTWARE SPECIFICATION:**

* Operating system : Windows 7,8.
* Front End : JAVA and XML
* Back End : SQL Server 2005

**FRONT END: JAVA**

**JAVA**

**Java** is a general-purpose [computer-programming language](https://en.wikipedia.org/wiki/Programming_language" \o "Programming language) that is [concurrent](https://en.wikipedia.org/wiki/Concurrent_computing" \o "Concurrent computing), [class-based](https://en.wikipedia.org/wiki/Class-based_programming" \o "Class-based programming), [object-oriented](https://en.wikipedia.org/wiki/Object-oriented_programming" \o "Object-oriented programming), and specifically designed to have as few implementation dependencies as possible. It is intended to let application developers "[write once, run anywhere](https://en.wikipedia.org/wiki/Write_once,_run_anywhere" \o "Write once, run anywhere)" (WORA),meaning that [compiled](https://en.wikipedia.org/wiki/Compiler" \o "Compiler) Java code can run on all platforms that support Java without the need for recompilation. Java applications are typically compiled to [bytecode](https://en.wikipedia.org/wiki/Java_bytecode" \o "Java bytecode) that can run on any [Java virtual machine](https://en.wikipedia.org/wiki/Java_virtual_machine" \o "Java virtual machine) (JVM) regardless of [computer architecture](https://en.wikipedia.org/wiki/Computer_architecture" \o "Computer architecture). As of 2016, Java is one of the most [popular programming languages in use](https://en.wikipedia.org/wiki/Measuring_programming_language_popularity" \o "Measuring programming language popularity), particularly for client-server web applications, with a reported 9 million developers. Java was originally developed by [James Gosling](https://en.wikipedia.org/wiki/James_Gosling" \o "James Gosling), a Canadian, at [Sun Microsystems](https://en.wikipedia.org/wiki/Sun_Microsystems" \o "Sun Microsystems) (which has since been [acquired by Oracle Corporation](https://en.wikipedia.org/wiki/Sun_acquisition_by_Oracle" \o "Sun acquisition by Oracle)) and released in 1995 as a core component of Sun Microsystems' [Java platform](https://en.wikipedia.org/wiki/Java_(software_platform)" \o "Java (software platform)). The language derives much of its original features from [SmallTalk](https://en.wikipedia.org/wiki/SmallTalk" \o "SmallTalk), with a [syntax](https://en.wikipedia.org/wiki/Syntax_(programming_languages)" \o "Syntax (programming languages)) similar to [C](https://en.wikipedia.org/wiki/C_(programming_language)" \o "C (programming language)) and [C++](https://en.wikipedia.org/wiki/C++" \o "C++), but it has fewer [low-level](https://en.wikipedia.org/wiki/Low-level_programming_language" \o "Low-level programming language) facilities than either of them.

* JAVA provides services to allow the creation, deployment, and execution of Web Applications and Web Services
* Like ASP, JAVA is a server-side technology
* Web Applications are built using Web Forms. JAVA comes with built-in Web Forms controls, which are responsible for generating the user interface. They mirror typical HTML widgets like text boxes or buttons. If these controls do not fit your needs, you are free to create your own user controls.
* Web Forms are designed to make building web-based applications as easy as building Visual Basic applications

**JAVA Architecture**

JAVA is based on the fundamental architecture of Framework. Visual studio provide a uniform way to combine the various features of this Architecture.

Architecture is explained from bottom to top in the following:

At the bottom of the Architecture is Common Language Runtime .NET Framework common language runtime resides on top of the operating system services. The common language runtime loads and executes code that targets the runtime. This code is therefore called managed code. The runtime gives you, for example, the ability for cross-language integration.

This is evident from its two major design principles:

1. Disconnected Datasets—In ADO.NET, almost all data manipulation is done outside the context of an open database connection.

2. Effortless Data Exchange with XML—Datasets can converse in the universal data format of the Web, namely XML.

The 4th layer of the framework consists of the Windows application model and, in parallel, the Web application model. The Web application model-in the slide presented as JAVA-includes Web Forms and Web Services.

JAVA comes with built-in Web Forms controls, which are responsible for generating the user interface. They mirror typical HTML widgets like text boxes or buttons. If these controls do not fit your needs, you are free to create your own user controls.

Web Services brings you a model to bind different applications over the Internet. This model is based on existing infrastructure and applications and is therefore Standard-based, simple, and adaptable.

Web Services are software solutions delivered via Internet to any device. Today, that means Web browsers on computers, for the most part, but the device-agnostic design of .NET will eliminate this limitation.

The CLR and the .NET Frameworks in general, however, are designed in such a way that code written in one language can not only seamlessly be used by another language. Hence JAVA can be programmed in any of the .NET compatible language whether it is VB.NET, C#, Managed C++ or JScript

### Java JVM and Byte code

Main articles: [Java (software platform)](https://en.wikipedia.org/wiki/Java_(software_platform)" \o "Java (software platform)) and [Java virtual machine](https://en.wikipedia.org/wiki/Java_virtual_machine" \o "Java virtual machine)

One design goal of Java is portability, which means that programs written for the Java platform must run similarly on any combination of hardware and operating system with adequate run time support. This is achieved by compiling the Java language code to an intermediate representation called [Java bytecode](https://en.wikipedia.org/wiki/Java_bytecode" \o "Java bytecode), instead of directly to architecture-specific [machine code](https://en.wikipedia.org/wiki/Machine_code" \o "Machine code). Java byte code instructions are analogous to machine code, but they are intended to be executed by a [virtual machine](https://en.wikipedia.org/wiki/Virtual_machine" \o "Virtual machine) (VM) written specifically for the host hardware. [End users](https://en.wikipedia.org/wiki/End_user" \o "End user) commonly use a [Java Runtime Environment](https://en.wikipedia.org/wiki/Java_virtual_machine" \o "Java virtual machine) (JRE) installed on their own machine for standalone Java applications, or in a web browser for Java [applets](https://en.wikipedia.org/wiki/Applet" \o "Applet).

Standard libraries provide a generic way to access host-specific features such as graphics, [threading](https://en.wikipedia.org/wiki/Thread_(computing)" \o "Thread (computing)), and [networking](https://en.wikipedia.org/wiki/Computer_network" \o "Computer network).

The use of universal byte code makes porting simple. However, the overhead of interpreting byte code into machine instructions made interpreted programs almost always run more slowly than native [executables](https://en.wikipedia.org/wiki/Executable" \o "Executable). [Just-in-time](https://en.wikipedia.org/wiki/Just-in-time_compilation" \o "Just-in-time compilation) (JIT) compilers that compile byte-codes to machine code during run time were introduced from an early stage. Java itself is platform-independent and is adapted to the particular platform it is to run on by a [Java virtual machine](https://en.wikipedia.org/wiki/Java_virtual_machine" \o "Java virtual machine) for it, which translates the [Java bytecode](https://en.wikipedia.org/wiki/Java_bytecode" \o "Java bytecode) into the platform's machine language.

#### Performance

Main article: [Java performance](https://en.wikipedia.org/wiki/Java_performance" \o "Java performance)

Programs written in Java have a reputation for being slower and requiring more memory than those written in C++. However, Java programs' execution speed improved significantly with the introduction of [just-in-time compilation](https://en.wikipedia.org/wiki/Just-in-time_compilation" \o "Just-in-time compilation) in 1997/1998 for [Java 1.1](https://en.wikipedia.org/wiki/Java_version_history" \o "Java version history), the addition of language features supporting better code analysis (such as inner classes, the String Builder class, optional assertions, etc.), and optimization in the Java virtual machine, such as [HotSpot](https://en.wikipedia.org/wiki/HotSpot" \o "HotSpot) becoming the default for Sun's JVM in 2000. With Java 1.5, the performance was improved with the addition of the java.util.concurrent package, including [lock free](https://en.wikipedia.org/wiki/Lock_free" \o "Lock free) implementations of the [ConcurrentMaps](https://en.wikipedia.org/wiki/Java_ConcurrentMap" \o "Java ConcurrentMap) and other multi-core collections, and it was improved further with Java 1.6.

### Non-JVM

Some platforms offer direct hardware support for Java; there are micro controllers that can run Java byte code in hardware instead of a software Java virtual machine, and some [ARM](https://en.wikipedia.org/wiki/ARM_architecture" \o "ARM architecture)-based processors could have hardware support for executing Java byte code through their [Jazelle](https://en.wikipedia.org/wiki/Jazelle" \o "Jazelle) option, though support has mostly been dropped in current implementations of ARM.

### Automatic memory management

Java uses an [automatic garbage collector](https://en.wikipedia.org/wiki/Garbage_collection_(computer_science)" \o "Garbage collection (computer science)) to manage memory in the [object lifecycle](https://en.wikipedia.org/wiki/Object_lifetime" \o "Object lifetime). The programmer determines when objects are created, and the Java run time is responsible for recovering the memory once objects are no longer in use. Once no references to an object remain, the [unreachable memory](https://en.wikipedia.org/wiki/Unreachable_memory" \o "Unreachable memory) becomes eligible to be freed automatically by the garbage collector. Something similar to a [memory leak](https://en.wikipedia.org/wiki/Memory_leak" \o "Memory leak) may still occur if a programmer's code holds a reference to an object that is no longer needed, typically when objects that are no longer needed are stored in containers that are still in use. If methods for a non-existent object are called, a "null pointer exception" is thrown.

One of the ideas behind Java's automatic memory management model is that programmers can be spared the burden of having to perform manual memory management. In some languages, memory for the creation of objects is implicitly allocated on the [stack](https://en.wikipedia.org/wiki/Stack_(abstract_data_type)" \o "Stack (abstract data type)) or explicitly allocated and deallocated from the [heap](https://en.wikipedia.org/wiki/Memory_management" \l "DYNAMIC" \o "Memory management). In the latter case, the responsibility of managing memory resides with the programmer. If the program does not deallocate an object, a [memory leak](https://en.wikipedia.org/wiki/Memory_leak" \o "Memory leak) occurs. If the program attempts to access or deallocate memory that has already been deallocated, the result is undefined and difficult to predict, and the program is likely to become unstable or crash. This can be partially remedied by the use of [smart pointers](https://en.wikipedia.org/wiki/Smart_pointer" \o "Smart pointer), but these add overhead and complexity. Note that garbage collection does not prevent "logical" memory leaks, *i.e.*, those where the memory is still referenced but never used.

Garbage collection may happen at any time. Ideally, it will occur when a program is idle. It is guaranteed to be triggered if there is insufficient free memory on the heap to allocate a new object; this can cause a program to stall momentarily. Explicit memory management is not possible in Java.

Java does not support C/C++ style [pointer arithmetic](https://en.wikipedia.org/wiki/Pointer_(computer_programming)" \o "Pointer (computer programming)), where object addresses can be arithmetically manipulated (e.g. by adding or subtracting an offset). This allows the garbage collector to relocate referenced objects and ensures type safety and security.

As in C++ and some other object-oriented languages, variables of Java's [primitive data types](https://en.wikipedia.org/wiki/Primitive_data_type" \o "Primitive data type) are either stored directly in fields (for objects) or on the [stack](https://en.wikipedia.org/wiki/Stack-based_memory_allocation" \o "Stack-based memory allocation) (for methods) rather than on the heap, as is commonly true for non-primitive data types (but see [escape analysis](https://en.wikipedia.org/wiki/Escape_analysis" \o "Escape analysis)). This was a conscious decision by Java's designers for performance reasons.

Java contains multiple types of garbage collectors. By default, HotSpot uses the [parallel scavenge garbage collector](https://en.wikipedia.org/w/index.php?title=Parallel_scavenge_garbage_collector&action=edit&redlink=1" \o "Parallel scavenge garbage collector (page does not exist)).[[57]](https://en.wikipedia.org/wiki/Java_(programming_language)" \l "cite_note-57) However, there are also several other garbage collectors that can be used to manage the heap. For 90% of applications in Java, the [Concurrent Mark-Sweep](https://en.wikipedia.org/wiki/Concurrent_mark_sweep_collector" \o "Concurrent mark sweep collector) (CMS) garbage collector is sufficient.[[58]](https://en.wikipedia.org/wiki/Java_(programming_language)" \l "cite_note-58) Oracle aims to replace CMS with the [Garbage-First collector](https://en.wikipedia.org/wiki/Garbage-First_collector" \o "Garbage-First collector) (G1).

Having solved the memory management problem does not relieve the programmer of the burden of handling properly other kind of resources, like network or database connections, file handles, etc., especially in the presence of exceptions. Paradoxically, the presence of a garbage collector has faded the necessity of having an explicit destructor method in the classes, thus rendering the management of these other resources more difficult!

**Web Services**

* Web services are small units of code built to handle a limited task.
* Web services uses XML based communicating protocols.
* Its operating systems are independent. And programming languages also independent.
* It’s connecting the people, system, and devices.
* Web services use the standard web protocols HTTP, XML, SOAP, WSDL, and UDDI.

**Benefits of Web Services**

* Easy to communicate between the application.
* Easy to reuse existing system.
* Rapid development.
* Easier to distribute information to more consumers.
* It can create new possibilities for many business because it provides an easy way to a large number of consumers.

**Important Features of JAVA**

**Robust database-driven Functionality**

JAVA is purely object-oriented makes it very powerful. JAVA allows developers to develop web applications that interact with the database.

JAVA (ADO stands for Activex Data Objects) is especially launched for JAVA for handling all types of database related queries.

**Faster Web Applications**

JAVA Web applications are faster due to two reasons: compiled code and caching. Earlier, the code was interpreted into machine language when website visitor used to view the page.

Caching is the storing of information in a non-physical media for speeding up the processing for the future use. Web pages that are commonly visited by end-users are cached for a predefined period of time to improve the performance of website.

**Memory Leak and Crash Protection**

Now developers need not bother about error like deadlocks and memory leaks because JAVA can automatically recover from these errors and your website always remains available to the end-users.

Memory leak is a situation in which a program component fails to release the memory when the memory is no longer needed it.

A deadlock is a situation where each out of the two or more processors is waiting for some other process in the group to finish first and thus none of the processors finishes ever.

**Easy Deployment**

Development of web applications in JAVA is quiet easy because the configuration information is of built-in type.

**Multiple development Language support**

Developers can write their code in different .NET languages for example Visual Basic, Visual C#, Visual J#, etc.

**Difference between ASP and JAVA  
ASP:**

1. ASP is Interpreted language based on scripting languages like Jscript or VBScript.  
  
2. ASP has Mixed HTML and coding logic.  
  
3. Limited development and debugging tools available.  
  
4 Limited OOPS support.  
  
5. Limited session and application state management.  
  
6. Poor Error handling system.  
  
7. No in-built support for XML.  
  
8. No fully distributed data source support.

**JAVA:**1. JAVA is supported by compiler and has compiled language support.  
  
2. Separate code and design logic possible.  
  
3. Variety of compilers and tools available including the Visual Studio.NET.  
  
4. Completely Object Oriented.  
  
5. Complete session and application state management.  
  
6. Full proof error handling possible.  
  
7. Full XML Support for easy data exchange.  
  
8. Fully distributed data source support.

**Reason for Selecting the JAVA**

* Web applications created with JAVA are easier to create, debug, and deploy because those tasks can all be performed within a single development environment Visual Studio .NET.
* Executable portions of a Web application compiled so they execute more quickly than interpreted scripts.
* On-the-fly updates of deployed Web applications without restarting the server.
* Access to the .NET Framework, which extends the Windows API.
* Use of the widely known Visual Basic programming language, which has been enhanced to fully support object-oriented programming.
* Introduction of the new Visual C# programming language, which provides a type-safe, object-oriented version of the C programming language.
* Automatic state management for controls on a Web page (called server controls) so that they behave much more like Windows controls.
* The ability to create new, customized server controls from existing controls.
* Built-in security through the Windows server or through other authentication/authorization methods.
* Integration with Microsoft ADO.NET to provide database access and database design tools from within Visual Studio .NET.
* Full support for Extensible Markup Language (XML), cascading style sheets (CSS), and other new and established Web standards.
* Built-in features for caching frequently requested Web pages on the server, localizing content for specific languages and cultures, and detecting browser capabilities.

**BACK END USED**

**Microsoft SQL Server used as back end tool.**

Microsoft SQL Server2005

Microsoft SQL Server is an application used to create computer databases for the Microsoft windows family of server operating system. It provides an environment used to generate databases that can access workstations, the web or other media such as a Personal Digital Assistant (PDA).

Microsoft SQL Server is a probably the most accessible and the most documentation enterprise database environment right now. The following is a list of the features provided in SQL Server 2005.

* User-defined functions
* Indexed views
* Distributed partitioned views
* Cascading RI constraints
* Multiple SQL Server instances
* XML support

User-Defined Functions

SQL Server has always provided the ability to store and execute SQL code routines via stored procedures. In addition, SQL Server has always supplied a number of built-in functions. Functions can be used almost anywhere. An expression can be specified in query.

SQL Server 2005introduces the long-awaited support for user-defined functions. User-defined functions can take zero or more input parameters and return a single value like the system-defined functions, or a table result.

Table valued functions can be used anywhere. Table or view expressions can be used in queries, and they can perform more complex logic than is allowed in a view.

Indexed Views

Views are often used to simplify complex queries, and they can contain joins and aggregate functions. However, in the past, queries against views were resolved to queries against the underlying base table, and any aggregates were recalculated each time you ran a query against the view.

Indexed views can improve performance for the following types of queries:

* Joins and aggregate that process many rows.
* Join and aggregation operations that performed frequently with in many queries.
* Decision support queries that rely on summarized, aggregated data that is infrequently updated.

Distributed partitioned views

SQL server 7.0 provided the ability to create partitioned views using the UNION ALL statement in a view definition. It was limited, however, in that all the tables had to reside within the same SQL Server where the view was defined. SQL Server 2005expands the ability to create partitioned views by allowing to horizontally partition tables across multiple SQL Servers.

The features helps to scale out one database server to multiple database servers, while making the data appear as if it comes from a single table on a single SQL Server. In addition, partitioned views are now able to be updated.

Cascading RI Constraints

In previous versions of SQL Server, referential integrity (RI) constraints were restrictive only. If an insert, update, or delete operation violated referential integrity, it was aborted with an error message.

SQL Server 2005 provides the ability to specify the action to take when a column referenced by a foreign key constrain is updated or deleted. You can still abort the update or delete if related foreign key records exist by specifying the NO ACTION option, or you can specify the new CASCADE option, which will cascade the update or delete operation to the related foreign key records.

Multiple SQL Server Instances

Previous versions of SQL Server supported the running of only a single instance of SQL Server at a time on a computer.

Running multiple instance or multiple versions of SQL Server required switching back and forth between the instances, requiring changes in the Windows registry. (The SQL Server Switch provides with 7.0 switches between 7.0 and 6.5 performed the registry changes for us.)

SQL Server 2005 provides support for running multiple instances of SQL Server on the same system. This allows you to simultaneously run one instance of SQL Server 6.5 or 7.0 along with one or more instances of SQL Server 2005.

Each SQL Server instance runs independently of the others and has its own set of system and user databases, security configuration and so on. Applications can connect to the different instance in the same way they connect to different SQL Servers on different machine.

XML Support

Extensible Markup Language has became a standard in web-related programming to describe the contents of a set of data and how the data should be output or displayed on a web page. XML, like HTML, is derived from the Standard Generalize Markup Language (SGML).

When linking a web application to SQL Server, a translation needs to take place from the result set returned from SQL Server to a format that can be understood and displayed web applications. Previously, this translation needed to be done in a client application.

SQL Server 2005 provides native support XML. These new features provide the ability to do the following:

Return query result sets directly in XML format.

* Retrieve data from an XML document as it were a SQL Server table.
* Access SQL Server through a URL using HTTP. Through Internet Information Services (IIS), you can define a virtual root that gives you HTTP access to the data and XML functionality of SQL Server 2000.

The features of the SQL Server 2005

It is a client-server architecture and not shared-file application as access.

* Symmetric Multiprocessing (SMP) supports up to 32 simultaneous processors.
* It can have database up to 1 terabyte (1024 GB) in size.
* It can handle up to 32,767 simultaneous user connections.
* It provides data integrity, data recovery and functionalities that are transparent to the developer.
* SQL Server optimizes network resources. Only the data requested by the client needs to be sent across the network.
* Security can be enforced at ones central location.

**Benefits of SQL**

* SQL has become the database language of choice because it is flexible, powerful and easy to learn.
* SQL is a non-procedural language.
* Process sets of records rather than just one at a time and provides automatic navigation to the data.
* SQL provides commands for a variety of tasks including: querying data, creating, updating and replacing and inserting, updating and deleting rows.
* All major relational database management systems support SQL thus one can transfer all the skills gained with SQL from one RDBMS to another.

**SYSTEM STUDY**

**EXISTING SYSTEM:**

The system which is used nowadays has some drawbacks which need to be improved for better performance. The system through which the feedback is taken is not good enough. The views of each and every student are not expressed through these systems. As the technology is developed day by day we need to use this technology so we can get an efficient result in adequate time. For attendance management in the present system all work is done on paper. The whole session attendance is stored in register and at the end of the session the reports are generated. We are not interested in generating report in the middle of the session or as per the requirement because it takes more time in calculation. At the end of session the students who don’t have

.

**DISADVANTAGES:**

The existing system has the following drawbacks.

* The existing system is not user friendly because the retrieval of data is very slow and data is not maintained efficiently.
* It require more calculations to generate the report so it is generated at the end of the session.
* All calculations to generate report is done manually so there is greater chance of errors. Here the faculty has to suffer a lot through the calculation and if there is a loss of some report then it may cause a lot of problem

**PROPOSED SYSTEM:**

The system architecture has a smart phone with android OS, a web services, a database server and the user as its components. The android smart phone or tablet must use 3G or WiFi network for internet connectivity to ensure better performance however 2G should also satisfy user request with added disadvantage of time lag. The user will login to the application through an android smart phone. The user-type is verified with the database server and access is given to the appropriate user. The web application also can be used to login and perform certain operations such as registration of users, generation of reports. The web application and the android application access data from a common Database server through the internet.

**ADVANTAGES OF THE PROPOSED SYSTEM:**

* It’s a user friendly user can easily understand this application.
* Reduce the unwanted issues.
* Parents easily identify his/her child studies details

**SYSTEM DESIGN AND DEVELOPMENT**

**INPUT DESIGN**

The input design is the link between the information system and the user. It comprises the developing specification and procedures for data preparation and those steps are necessary to put transaction data in to a usable form for processing can be achieved by inspecting the computer to read data from a written or printed document or it can occur by having people keying the data directly into the system. The design of input focuses on controlling the amount of input required, controlling the errors, avoiding delay, avoiding extra steps and keeping the process simple. The input is designed in such a way so that it provides security and ease of use with retaining the privacy. Input Design considered the following things:’

* What data should be given as input?
* How the data should be arranged or coded?
* The dialog to guide the operating personnel in providing input.
* Methods for preparing input validations and steps to follow when error occur.

**OBJECTIVES**

* Input Design is the process of converting a user-oriented description of the input into a computer-based system. This design is important to avoid errors in the data input process and show the correct direction to the management for getting correct information from the computerized system.
* It is achieved by creating user-friendly screens for the data entry to handle large volume of data. The goal of designing input is to make data entry easier and to be free from errors. The data entry screen is designed in such a way that all the data manipulates can be performed. It also provides record viewing facilities.
* When the data is entered it will check for its validity. Data can be entered with the help of screens. Appropriate messages are provided as when needed so that the user
* will not be in maize of instant. Thus the objective of input design is to create an input layout that is easy to follow

**OUTPUT DESIGN**

A quality output is one, which meets the requirements of the end user and presents the information clearly. In any system results of processing are communicated to the users and to other system through outputs. In output design it is determined how the information is to be displaced for immediate need and also the hard copy output. It is the most important and direct source information to the user. Efficient and intelligent output design improves the system’s relationship to help user decision-making.

1. Designing computer output should proceed in an organized, well thought out manner; the right output must be developed while ensuring that each output element is designed so that people will find the system can use easily and effectively. When analysis design computer output, they should Identify the specific output that is needed to meet the requirements.

2. Select methods for presenting information.

3. Create document, report, or other formats that contain information produced by the system.

The output form of an information system should accomplish one or more of the following objectives.

* Convey information about past activities, current status or projections of the
* Future.
* Signal important events, opportunities, problems, or warnings.
* Trigger an action.
* Confirm an action.

**MODULES**

1. Student
2. Staff
3. Feedback
4. Attendance
5. Home work

**MODULES DESCRIPTION:**

**Student**

In this module we are authenticating the users by providing username and password. If user name and password is valid then they will be taken to their screens.

**Staff**

This module is designed for staff, which use mobile phone to take attendance, upload result and upload college notifications. The entered admin details are encrypted and sent to server for verification. Only after successful authentication the operations are performed. If user name and password cannot match, he/she can enter in to attendance page.

**Feedback**

This module performs automated calculation of student’s feedback of their professors, and then this will be represent graphically. Student can see only fifteen questions and give feedback by select provided options. This module, stores student feedback into JSON object to provide faster transfer of feedback data to server or client-side. Next we extract string from the JSON to apply algorithm over and decide the result of feedback.

**Attendance**

The purpose of Attendance Entry Module is to enter the attendance using cell phone. In this module Lecturer takes the attendance using the cell phone. Lecturers select the branch, semester and year. After this session he enters in to attendance page. Here staff makes a mark on the absentees. Lecturer are only allowed to take attendance during their lecture time. In case of swapping of lecture or extra lecture, HOD can allow other faculty to take attendance of students by swapping the logic of time-table permanently or temporarily.

**Home work**

The purpose of home work module is an teacher give the home work to the class students via android application. Suppose teacher forgot to give a home then the teacher updating the home work then student or parents are notified.

**SYSTEM IMPLEMENTATION**

When the initial design was done for the system, the client was consulted for the acceptance of the design so that further proceedings of the system development can be carried on. After the development of the system a demonstration was given to them about the working of the system. The aim of the system illustration was to identify any malfunction of the system.

After the management of the system was approved the system implemented in the concern, initially the system was run parallel with existing manual system. The system has been tested with live data and has proved to be error free and user friendly.

Implementation is the process of converting a new or revised system design into an operational one when the initial design was done by the system; a demonstration was given to the end user about the working system.

This process is uses to verify and identify any logical mess working of the system by feeding various combinations of test data. After the approval of the system by both end user and management the system was implemented.

System implementation is made up of many activities. The six major activities are as follows.

**1. CODING**

Coding is the process of whereby the physical design specifications created by the analysis team turned into working computer code by the programming team. A design code may be a tool which helps ensure that the aspiration for quality and quantity for customers and their requirements, particularly for large scale projects, sought by the water agency Design pattern are documented tried and tested solutions for recurring problems in a given context. So basically you have a problem context and the proposed solution for the same.

**2. INSTALLATION**

Installation is the process during which the current system is replaced by the new system. This includes conversion of existing data, software, and documentation and work procedures to those consistent with the new system.

**3. DOCUMENTATION**

Documentation is descriptive information that describes the use and operation of the system. The user guide is provided to the end user as the student and administrator. The documentation part contains the details as follows,

User requirement and water agency details administration has been made online. Any customer can request their water requirement details through online and also use of documentation, they can view the purpose of each purpose, The admin could verify the authentication of the users, users requirements and need to take delivery process, thus the documentation is made of full view of project thus it gives the guideline to study the project and how to execute also.

**USER TRAINING AND SUPPORT**

The software is installed at the deployment environment, the developer will give training to the end user of the regional transport officer and police admin officer in that software. The goal of an end user training program is to produce a motivated user who has the skills needed to apply what has been to apply what has been learned to perform the job related task. The following are the instruction which is specified the handling and un-handling events in the application,

* The authenticated user of admin and office workers only login in the application with authorized username and password.
* Don’t make user waste their time to come straight to the water agency or make a phone call.
* It can easily track through online by the user.
* Very user friendliness software

**INSTALLATION STEPS**

Installation is the process during which the current system is replaced by the new system. This includes conversion of existing data, software, and documentation and work procedures to those consistent with the new system. The following steps are used to install the application in the end users’ machine.

Step 1: A folder named ‘SMART COLLEGE SYSTEM(main folder) is created in ‘D’ drive; but in any drive with any name the folder can created since the coding inside the application creates path dynamically in places wherever required.

Step 2: Two folders named ‘bin’ and ‘obj’ is created inside the main folder. Inside those two folders, a folder with name ‘Debug’ is created. All the files in the development machine’s Debug folder content are copied into this folder.

Step 3: The database folder is also copied to the main folder which contains the files ‘SMART COLLEGE SYSTEM \_Data.MDF’ and ‘SMART COLLEGE SYSTEM \_Log.LDF’.

Step 4: In Start->Programs->Microsoft SQL Server->Enterprise Manager Option is selected.

Step 5: The left side tree ‘Console Root’ is expanded until the local SQL Server instance node is open.

Step 6: Inside which the folder with name ‘Database’ is right clicked and All Tasks->Attach Database option is selected.

Step 7: The ellipsis button right side to the text box (for path) is clicked and the database main file ‘SMART COLLEGE SYSTEM\_Data.MDF’ is selected. Then OK button is clicked.

Step 8: After the message display about the successful database attachment, the executable file inside the D:\ SMART COLLEGE SYSTEM \bin\debug folder is double clicked and checked with one of the username and password input.

Step 9: The database is connected and so the main form appeared. This action confirms the application is running and as well as database connection is proper.

Step 10: The further options are worked out and checked so that all the records are appended in all the tables correctly.

**SYSTEM TESTING AND IMPLEMENTATION**

**SYSTEM TESTING**

System testing is the process of exercising software with the intent of finding and ultimately correcting errors. This fundamental philosophy does not change for web applications, because Web-based systems and application reside on a network and interoperate with many different operating system, browsers, hardware platforms, and communication protocols; the search for errors represents a significant challenge for web application.

The distributed nature of client\server environments, the performance issues associated with transaction processing, the potential presence of a number of different hardware platforms, the complexities of network communication, the need to serve multiple clients from a centralized database and the requirements imposed on the server all combine to make testing of client\server architectures.

System testing is actually a series of different tests whose primary purpose is to fully exercise the computer based system. System testing is the state of implementation that is aimed at assuring that the system works accurately and efficiently. Testing is the vital to the success of the system. System testing makes the logical assumption that if all the parts of the system are correct, the goal will be successfully achieved.

**The objective of testing is as follows:**

* + Testing is the process of executing a program with the intent of finding an error.
  + A successful test is that one of the cover of undiscovered error.

### TESTING ISSUES

* Client GUI considerations
* Target environment and platform diversity considerations
* Distributed database considerations
* Distributed processing considerations

**TESTING METHODOLOGIES**

System testing is state of implementation, which is aimed at ensuring that the system works accurately and efficiently as expect before live operation commences. It certifies that the whole set of programs hang together.

System testing requires a test plan that consists of several key activities and step for run program, string, system and user acceptance testing. The implementation of newly designed package is important in adopting a successful new system

Testing is the important stage in software development. the system test in implementation stage in software development process. The system testing implementation should be confirmation that all is correct and an opportunity to show the users that the system works as expected. It accounts the largest percentage of technical effort in the software development process.

Testing phase in the development cycle validates the code against the functional specification testing is vital to achievement of the system goals. The objective of the testing is to discover errors to fulfills this objective a series of test step unit, integration. validation and system tests were planned and executed the test steps are:

**System Testing**

Testing is an important phase in project development. System testing makes a logical assumption that if all parts of the system are correct, and the goal will be achieved successfully. The software must meet the user specification and it must satisfy according to the needs of the users.

Testing is the process of executing a project within the intend of finding errors. A good test case is one that has a high probability of finding an undiscovered error.

**Unit Testing**

Unit testing focuses verification efforts on the smallest unit of software design of the module. This is also known as “module testing”. This testing is carried out during programming stage itself. In this testing step, each module is found to be working satisfactorily as regards to the expected output of the modules.

**In Project**, Each module such customer registration module, request module, employee details module, stock module, vehicle module and area detail modules are tested individually for example, Customer details module can contain the more forms to maintain the information so all forms could be tested like entered information store appropriately in database access page or not. If correctly accessed means the testing of registration module successfully completed. Likewise all modules are tested successfully.

**Integration Testing**

Data can be lost across an interface, one module can have adverse effect on another sub function when combined it may not produce the desired major functions. Integration testing is a systematic testing for constructing test to uncover errors associated within an interface.

The objectives taken from unit tested modules and a program structure is built for integrated testing. All the modules are combined and the test is made.

A correction made in this testing is difficult because the vast expenses of the entire program complicated the isolation of causes. In this integration testing step, all the errors are corrected for next testing process.

**In Project,** Integration of two modules can be tested together such as customer registration details and customer login module for verification purposes providing proper accessibility to users. The communication of Registration and Login module can test and executed successfully.

**Validation Testing**

After the completion of the integrated testing, software is completely assembled as a package; interfacing error has been uncovered and corrected and a final series of software test validation begins.

Validation testing can be defined in many ways but a simple definition is that validation succeeds when the software function in a manner that can be reasonably expected by the customer. After validation test has been conducted, one of two possible conditions exists:

**In this project,** Admin login details form Enter without username and password in textbox enter the submit button then Login failed message otherwise checks the both textbox value that is true means valid page displayed. Enter Password Displaying password character \*.if it displays the characters security is not availed so testing of software is failed.

**Output Testing**

The next process of validation testing, is output testing of the proposed system, since no system could be successful if it does not produce the required output in the specified format. Asking the user about the format required, list the output to be generated or displayed by the system under considerations.

Output testing is a different test whose primary purpose is to fully exercise the computer based system although each test has a different purpose all the work should verify that all system elements have been properly integrated and perform allocated functions.

The output format on the screen is found to be corrected as the format was designed in the system design phase according to the user needs for the hard copy also; the output testing has not resulted in any correction in the system.

**In project** All the forms are tested as it gives the necessary output to the user’s search such as view response details.

**4.3 SYSTEM IMPLEMENTATION**

System implementation is the stage of the project that the theoretical design is turned into a working system. If the implementation stage is not properly planned and controlled, it can cause error. Thus it can be considered to be the most crucial stage in achieving a successful new system and in giving the user confidence that the new system will work and be effective.

Normally this stage involves setting up a coordinating committee, which will act as a sounding board for ideas; complaints and problem. The first task is implementation planning; i.e., deciding on the methods and time scale to be adopted. Apart from planning two major task of preparing for implementation are, education takes place much earlier in the project; at the implementation stage the emphasis must be on training in new skills to give staff confidence they can use the system. Once staff has been trained, the system can be tested.

After the implementation phase is completed and the user staff is adjusted to the changes created by the candidate system, evaluation and maintenance is to bring the new system to standards. The activities of the implementation phase can be summarized as,

* + - * Implementation planning
      * Education planning
      * System planning

**IMPLEMENTATION PROCEDURES**

Implementation includes all the activities that take place to convert the old system to the new one. Proper implementation is essential to provide a reliable system to meet the organization requirements. Implementation is the stage in the project where the theoretical design is turned into a working system. The most crucial stage is achieving a successful new system & giving the user confidence in that the new system will work efficiently & effectively in the implementation state.

**Implementation Procedures**

**Pilot Running:**

Processing the current data by only one user at a time called the pilot running process. When one user is accessing the data at one system, the system is sets to be engaged and connected in network. This process is useful only in system where more then one user is restricted.

**Parallel Running:**

Processing the current data by more then one user at a time simultaneously is said to be parallel running process. This same system can be viewed and accessed by more then one user at the time. Hence the implementation method used in the system is a pilot type of implementation.

Implementation is the stage in the project where the theoretical design is turned into a working system. The most crucial stage is achieving a successful new system & giving the user confidence in that the new system will work efficiently & effectively in the implementation state.

The stage consists of,

* Testing the developed program with sample data.
* Detection’s and correction of error.
* Creating whether the system meets user requirements.
* Making necessary changes as desired by the user.
* Training user personnel.

**4.3.2 USER MANUAL**

**User Training**

User Training is designed to prepare the user for testing &consenting the system. .

They are

1) User Manual.

2) Help Screens.

3) Training Demonstration.

**1) User Manual:**

The summary of important functions about the system and software can be provided as a document to the user.

**2) Help Screens:**

This features now available in every software package, especially when it is used with a menu. The user selects the “Help” option from the menu. The system accesses the necessary description or information for user reference.

**3) Training Demonstration:**

Another User Training element is a Training Demonstration. Live demonstrations with personal contact are extremely effective for Training Users.

**SYSTEM MAINTENANCE**

Maintenance is actually the implementation of the review plan. As important as it is, many programmers and analysts are to perform or identify themselves with the maintenance effort. There are psychological, personality and professional reasons for this. Analysts and programmers spend far more time maintaining programs than they do writing them. Maintenance accounts for 50-80 percent of total system development

Maintenance is expensive. One way to reduce the maintenance costs are through maintenance management and software modification audits***.***

* Maintenance is not as rewarding as exciting as developing systems. It is perceived as requiring neither skill not experience.
* Users are not fully cognizant of the maintenance problem or its high cost.
* Few tools and techniques are available for maintenance.
* A good test plan is lacking.
* Standards, procedures, and guidelines are poorly defined and enforced.
* Programs are often maintained without care for structure and documentation.
* There are minimal standards for maintenance.
* Programmers expect that they will not be in their current commitment by time their programs go into the maintenance cycle.

**Corrective Maintenance**

It means repairing, processing or performance failure or making changes because of previously uncovered problems or false assumptions. Task performed to identify, isolate, and rectify a fault so that the failed equipment, machine, or system can be restored to an operational condition within the tolerances or limits established for in-service operations.

Corrective maintenance can be subdivided into "immediate corrective maintenance" (in which work starts immediately after a failure) and "deferred corrective maintenance" (in which work is delayed in conformance to a given set of maintenance rules).

**Perfective Maintenance**

It means changes made to a system to add new features or to improve performance. Preventive maintenance is predetermined work performed to a schedule with the aim of preventing the wear and tear or sudden failure of equipment components. process or control equipment failure can have adverse results in both human and economic terms. In addition to down time and the costs involved to repair and/or replace equipment parts or components, there is the risk of injury to operators, and of acute exposures to chemical and/or physical agents.

Time-based or run-based Periodically inspecting, servicing, cleaning, or replacing parts to prevent sudden failure .On-line monitoring of equipment in order to use important/expensive parts to the limit of their serviceable life. Preventive maintenance involves changes made to a system to reduce the chance of future system failure.

An example of preventive maintenance might be to increase the number of records that a system can process far beyond what is currently needed or to generalize how a system sends report information to a printer so that so that the system can adapt to changes in printer technology.

**Preventive Maintenance**

Changes made to a system to avoid possible future problems Perfective maintenance involves making enhancements to improve processing performance, interface usability, or to add desired, but not necessarily required, system features. The objective of perfective maintenance is to improve response time, system efficiency, reliability, or maintainability.

  During system operation, changes in user activity or data pattern can cause a decline in efficiency, and perfective maintenance might be needed to restore performance. Usually, the perfective maintenance work is initiated by the IT department, while the corrective and adaptive maintenance work is normally requested by users.

**CONCLUSION AND FUTURE ENHANCEMENT**

**CONCLUSION:**

This project has been designed to complete the requirement of the school system is an completely convert to mobile application. Which is an very smart way to run an organization. A whole process student and faculty can using an mobile application and parents can also noticed their child via application. Which may very help to very user friendly so every people can easily using this application.

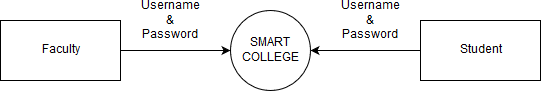
**FUTURE ENHANCEMENT**

This application is developed by using JAVA and MS SQL SERVER as back end for android development. In future this system may be developed by IOS or any other technology which is peak in that time. According to the user requirement every new technology or any techniques are developed. In future the security will be enhanced with high level. A New technology is implemented with this system to get better performance. The future system will be developed with different features according to the users need. The future enhancement includes a more advance features and secures the data which will be incorporate all the methods, which are convenient for the organization to give the better performance.

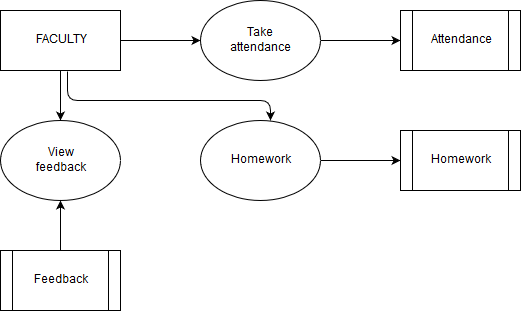
**APPENDIX**

**DATA FLOW DIAGRAM**

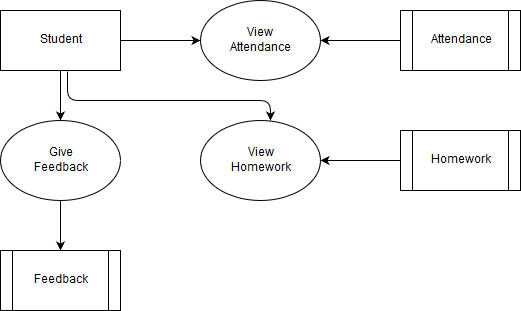
**LEVEL 0:**

****

**LEVEL 1:**

****

LEVEL 2:



**TABLES:**

**Table Name :** class

**Primary Key :** id

**Foreign Key :** null

**Description :** Class List Table

|  |  |  |  |
| --- | --- | --- | --- |
| **FIELD NAME** | **DATA TYPE** | **SIZE** | **DESCRIPTION** |
| ID | INT | 10 | CLASS ID |
| CLASS\_NAME | VARCHAR | 20 | CLASS NAME |

**Table Name :** user

**Primary Key :** id

**Foreign Key :** cid

**Description :** User List Table

|  |  |  |  |
| --- | --- | --- | --- |
| **FIELD NAME** | **DATA TYPE** | **SIZE** | **DESCRIPTION** |
| ID | INT | 10 | USER ID |
| CID | INT | 10 | CLASS ID |
| U\_TYPE | VARCHAR | 1 | USER TYPE |
| USERNAME | VARCHAR | 30 | USERNAME |
| PASSWORD | VARCHAR | 30 | PASSWORD |
| NAME | VARCHAR | 30 | USER NAME |
| MOBILE | VARCHAR | 10 | MOBILE |
| AGE | INT | 3 | AGE |
| DOB | DATE | 15 | DATE OF BIRTH |
| EMAIL | VARCHAR | 30 | EMAIL ID |

**TABLES :**

**Table Name :** Attendance Table

**Primary Key :** id

**Foreign Key :** sid , cid

**Description :** Attendance Table

|  |  |  |  |
| --- | --- | --- | --- |
| **FIELD NAME** | **DATA TYPE** | **SIZE** | **DESCRIPTION** |
| ID | INT | 10 | ID |
| SID | INT | 10 | STUDENT ID |
| CID | INT | 10 | CLASS ID |
| DATE | DATE | 10 | DATE |
| I | VARCHAR | 1 | HOUR 1 |
| II | VARCHAR | 1 | HOUR 2 |
| III | VARCHAR | 1 | HOUR 3 |
| IV | VARCHAR | 1 | HOUR 4 |
| V | VARCHAR | 1 | HOUR 5 |
| VI | VARCHAR | 1 | HOUR 6 |

**TABLES :**

**Table Name :** home work

**Primary Key :** id

**Foreign Key :** cid,given

**Description :** Home work Table

|  |  |  |  |
| --- | --- | --- | --- |
| **FIELD NAME** | **DATA TYPE** | **SIZE** | **DESCRIPTION** |
| ID | INT | 10 | ID |
| CID | INT | 10 | CLASS ID |
| SUB | TEXT | 30 | SUBJECT |
| DES | TEXT | 30 | DESCRIPTION |
| GIVEN | INT | 10 | GIVEN BY |
| DATE | DATE | 10 | DATE |

**TABLES :**

**Table Name :** feedback

**Primary Key :** id

**Foreign Key :** null

**Description :** Feedback Table

|  |  |  |  |
| --- | --- | --- | --- |
| **FIELD NAME** | **DATA TYPE** | **SIZE** | **DESCRIPTION** |
| ID | INT | 10 | ID |
| CATEGORY | VARCHAR | 30 | CATEGORY |
| DES | TEXT | 30 | DESCRIPTION |
| DATE | DATE | 10 | DATE |

SAMPLE CODING

package com.kgfsl.ecampusstudent.Activity;

import android.support.v7.app.AppCompatActivity;

import android.os.Bundle;

import android.support.v7.widget.LinearLayoutManager;

import android.support.v7.widget.RecyclerView;

import android.widget.Toast;

import com.android.volley.Request;

import com.android.volley.RequestQueue;

import com.android.volley.Response;

import com.android.volley.VolleyError;

import com.android.volley.toolbox.StringRequest;

import com.android.volley.toolbox.Volley;

import com.kgfsl.ecampusstudent.ClassListAdapter;

import com.kgfsl.ecampusstudent.R;

import com.kgfsl.ecampusstudent.URL;

import org.json.JSONArray;

import org.json.JSONException;

import org.json.JSONObject;

import java.util.ArrayList;

public class ClassListActivity extends AppCompatActivity {

RecyclerView recyclerView;

LinearLayoutManager linearLayoutManager;

String FLAG,id;

URL url;

ClassListAdapter listAdapter;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_class\_list);

FLAG = getIntent().getStringExtra("FLAG");

id = getIntent().getStringExtra("id");

recyclerView=(RecyclerView)findViewById(R.id.recyclerview);

linearLayoutManager=new LinearLayoutManager(getApplicationContext(),LinearLayoutManager.VERTICAL,false);

recyclerView.setLayoutManager(linearLayoutManager);

getClassList();

}

private void getClassList() {

final ArrayList<String> idList=new ArrayList<>();

final ArrayList<String> cnameList=new ArrayList<>();

final ArrayList<ClassModel> data=new ArrayList<>();

RequestQueue requestQueue = Volley.newRequestQueue(this);

StringRequest stringRequest = new StringRequest(Request.Method.GET, url.classlist, new Response.Listener<String>() {

@Override

public void onResponse(String response) {

try {

JSONObject jsonObject = new JSONObject(response);;

JSONArray jsonArray = jsonObject.getJSONArray("data");

for(int i=0;i<jsonArray.length();i++){

JSONObject jsonObject1 = jsonArray.getJSONObject(i);

String id = jsonObject1.getString("id");

String cls = jsonObject1.getString("class");

idList.add(id);

cnameList.add(cls);

data.add(new ClassModel(id,cls));

}

listAdapter = new ClassListAdapter(ClassListActivity.this,data,FLAG,id);

recyclerView.setAdapter(listAdapter);

} catch (JSONException e) {

toast(String.valueOf(e));

e.printStackTrace();

}

}

}, new Response.ErrorListener() {

@Override

public void onErrorResponse(VolleyError error) {

toast(String.valueOf(error));

}

});

requestQueue.add(stringRequest);

}

private void toast(String s) {

Toast.makeText(getApplicationContext(),s,Toast.LENGTH\_SHORT).show();

}

}

package com.kgfsl.ecampusstudent.Activity;

import android.content.Intent;

import android.support.v7.app.AppCompatActivity;

import android.os.Bundle;

import android.view.View;

import android.widget.LinearLayout;

import com.kgfsl.ecampusstudent.AboutUsActivity;

import com.kgfsl.ecampusstudent.ProfileActivity;

import com.kgfsl.ecampusstudent.R;

public class TeacherHomeActivity extends AppCompatActivity implements View.OnClickListener {

LinearLayout attendance,homework,profile,feedback,about,logout;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_teacher\_home);

attendance = (LinearLayout)findViewById(R.id.teacher\_attendance);

homework = (LinearLayout)findViewById(R.id.teacher\_homework);

profile = (LinearLayout)findViewById(R.id.teacher\_profile);

feedback = (LinearLayout)findViewById(R.id.teacher\_feedback);

about = (LinearLayout)findViewById(R.id.teacher\_about);

logout = (LinearLayout)findViewById(R.id.teacher\_logout);

attendance.setOnClickListener(this);

profile.setOnClickListener(this);

homework.setOnClickListener(this);

feedback.setOnClickListener(this);

about.setOnClickListener(this);

logout.setOnClickListener(this);

}

@Override

public void onClick(View v) {

switch (v.getId()){

case R.id.teacher\_profile:

Intent pro = new Intent(TeacherHomeActivity.this,ProfileActivity.class);

pro.putExtra("FLAG","Attendance");

pro.putExtra("id",getIntent().getStringExtra("id"));

startActivity(pro);

break;

case R.id.teacher\_attendance:

Intent intent = new Intent(TeacherHomeActivity.this,ClassListActivity.class);

intent.putExtra("FLAG","Attendance");

intent.putExtra("id",getIntent().getStringExtra("id"));

startActivity(intent);

break;

case R.id.teacher\_homework:

Intent home = new Intent(TeacherHomeActivity.this,ClassListActivity.class);

home.putExtra("FLAG","homework");

home.putExtra("id",getIntent().getStringExtra("id"));

startActivity(home);

break;

case R.id.teacher\_about:

Intent about = new Intent(TeacherHomeActivity.this,AboutUsActivity.class);

startActivity(about);

break;

case R.id.teacher\_feedback:

Intent feedback = new Intent(TeacherHomeActivity.this,FeedbackActivity.class);

startActivity(feedback);

break;

case R.id.teacher\_logout:

Intent log = new Intent(TeacherHomeActivity.this,LoginActivity.class);

log.addFlags(Intent.FLAG\_ACTIVITY\_CLEAR\_TOP);

startActivity(log);

break;

}

}

}

package com.kgfsl.ecampusstudent.Fragment;

import android.annotation.SuppressLint;

import android.os.Bundle;

import android.support.v4.app.Fragment;

import android.support.v7.widget.LinearLayoutManager;

import android.support.v7.widget.RecyclerView;

import android.view.LayoutInflater;

import android.view.View;

import android.view.ViewGroup;

import android.widget.Toast;

import com.android.volley.Request;

import com.android.volley.RequestQueue;

import com.android.volley.Response;

import com.android.volley.VolleyError;

import com.android.volley.toolbox.StringRequest;

import com.android.volley.toolbox.Volley;

import com.fasterxml.jackson.databind.ObjectMapper;

import com.kgfsl.ecampusstudent.Adapter.Attendance.AttendanceAdapterDate;

import com.kgfsl.ecampusstudent.Adapter.Attendance.AttendanceAdapterMonth;

import com.kgfsl.ecampusstudent.Pojo.Attendance.AttendanceDate;

import com.kgfsl.ecampusstudent.Pojo.Attendance.AttendanceMonth;

import com.kgfsl.ecampusstudent.R;

import com.kgfsl.ecampusstudent.URL;

import java.io.IOException;

import java.util.HashMap;

import java.util.Map;

/\*\*

\* A simple {@link Fragment} subclass.

\*/

public class AttendanceFragment extends Fragment {

View v;

RecyclerView hRV, vRV;

String id;

public AttendanceFragment() {

}

@SuppressLint("ValidFragment")

public AttendanceFragment(String id) {

this.id=id;

}

@Override

public View onCreateView(LayoutInflater inflater, ViewGroup container,

Bundle savedInstanceState) {

v = inflater.inflate(R.layout.fragment\_attendance, container, false);

initialize();

return v;

}

private void initialize() {

hRV = (RecyclerView) v.findViewById(R.id.horzon\_recyclerview);

vRV = (RecyclerView) v.findViewById(R.id.vertical\_recyclerview);

LinearLayoutManager hLinearLayoutManager = new LinearLayoutManager(v.getContext(), LinearLayoutManager.HORIZONTAL, false);

LinearLayoutManager vLinearLayoutManager = new LinearLayoutManager(v.getContext(), LinearLayoutManager.VERTICAL, false);

hRV.setLayoutManager(hLinearLayoutManager);

vRV.setLayoutManager(vLinearLayoutManager);

getAttendance();

}

@Override

public void onStart() {

super.onStart();

getAttendance();

}

public void getAttendance() {

String json = "{\n" +

" \"status\":true,\n" +

" \"Sem\":[\"II\"],\n" +

" \"months\":[\"JAN\",\"FEB\",\"MAR\",\"APR\",\"MAY\",\"JUN\",\"JUL\",\"AUG\",\"SEP\",\"OCT\",\"NOV\",\"DEC\"]\n" +

"}";

ObjectMapper objectMapper = new ObjectMapper();

try {

AttendanceMonth response = objectMapper.readValue(json, AttendanceMonth.class);

if (response.getStatus()) {

AttendanceAdapterMonth attendanceAdapterMonth = new AttendanceAdapterMonth(v.getContext(), response, new AttendanceAdapterMonth.OnClickListner() {

@Override

public void MonthClickListner(AttendanceMonth attendanceMonth, int position) {

AttendanceDate(attendanceMonth.getMonths().get(position));

}

});

// hRV.setAdapter(attendanceAdapterMonth);

// attendanceAdapterMonth.notifyDataSetChanged();

AttendanceDate("");

}

} catch (IOException e) {

e.printStackTrace();

toast(String.valueOf(e));

}

}

private void AttendanceDate(String month) {

String json="{\"status\":true,\"punchcount\":6,\"Dates\":[{\"date\":\"10-10-2017\",\"day\":\"Monday\",\"dayStatus\":\"W\",\"att\_report\":[\"P\",\"P\",\"A\",\"P\",\"P\",\"A\"]},{\"date\":\"11-10-2017\",\"day\":\"Tuesday\",\"dayStatus\":\"W\",\"att\_report\":[\"P\",\"P\",\"P\",\"P\",\"P\",\"P\"]},{\"date\":\"12-10-2017\",\"day\":\"Wednesday\",\"dayStatus\":\"W\",\"att\_report\":[\"P\",\"P\",\"P\",\"A\",\"P\",\"P\"]},{\"date\":\"13-10-2017\",\"day\":\"Thursday\",\"dayStatus\":\"W\",\"att\_report\":[\"P\",\"P\",\"A\",\"P\",\"P\",\"A\"]},{\"date\":\"14-10-2017\",\"day\":\"Friday\",\"dayStatus\":\"W\",\"att\_report\":[\"A\",\"P\",\"P\",\"P\",\"P\",\"P\"]},{\"date\":\"15-10-2017\",\"day\":\"Saturday\",\"dayStatus\":\"W\",\"att\_report\":[\"P\",\"A\",\"P\",\"A\",\"P\",\"P\"]},{\"date\":\"17-10-2017\",\"day\":\"Monday\",\"dayStatus\":\"W\",\"att\_report\":[\"P\",\"P\",\"A\",\"P\",\"P\",\"A\"]},{\"date\":\"18-10-2017\",\"day\":\"Tuesday\",\"dayStatus\":\"W\",\"att\_report\":[\"P\",\"P\",\"A\",\"P\",\"P\",\"P\"]},{\"date\":\"19-10-2017\",\"day\":\"Wednesday\",\"dayStatus\":\"W\",\"att\_report\":[\"A\",\"P\",\"A\",\"A\",\"P\",\"A\"]}]}";

RequestQueue requestQueue = Volley.newRequestQueue(v.getContext());

StringRequest stringRequest = new StringRequest(Request.Method.POST, URL.attendancedetails, new Response.Listener<String>() {

@Override

public void onResponse(String response1) {

ObjectMapper objectMapper = new ObjectMapper();

try {

AttendanceDate response = objectMapper.readValue(response1, AttendanceDate.class);

if (response.getStatus()) {

AttendanceAdapterDate attendanceAdapterDate = new AttendanceAdapterDate(v.getContext(),response);

vRV.setAdapter(attendanceAdapterDate);

attendanceAdapterDate.notifyDataSetChanged();

}

} catch (IOException e) {

e.printStackTrace();

toast(String.valueOf(e));

}

}

}, new Response.ErrorListener() {

@Override

public void onErrorResponse(VolleyError error) {

toast(String.valueOf(error));

}

}){

@Override

protected Map<String, String> getParams()

{

Map<String, String> params = new HashMap<String, String>();

params.put("id", id);

return params;

}

};

requestQueue.add(stringRequest);

}

private void toast(String toast) {

Toast.makeText(v.getContext(), toast, Toast.LENGTH\_SHORT).show();

}

}

package com.kgfsl.ecampusstudent;

/\*\*

\* Created by Madhan on 25-12-2017.

\*/

public class URL {

//public final static String URL="http://192.168.43.153/gokul/api/api/college";

public final static String URL="http://192.168.43.127/gokul/college";

//public final static String URL="http://172.30.3.45/gokul/college";

public final static String register =URL+"/register.php";

public final static String login=URL+"/login.php";

public final static String profile=URL+"/profile.php?id=";

public final static String alldonorslist=URL+"/alldonorslist.php?id=";

public final static String sending\_request=URL+"/sendingrequest.php";

public final static String classlist=URL+"/classlist.php";

public final static String studentlist=URL+"/studentllist.php";

public final static String attendanceregister=URL+"/attendanceregister.php";

public final static String attendancedetails=URL+"/attendancedetails.php";

public final static String putHomework=URL+"/puthomework.php";

public final static String homeworklist=URL+"/viewhomework.php";

public final static String postfeedback=URL+"/postfeedback.php";

public final static String viewfeedback=URL+"/viewfeedback.php";

}

package com.kgfsl.ecampusstudent;

import android.app.AlertDialog;

import android.content.DialogInterface;

import android.support.v7.app.AppCompatActivity;

import android.os.Bundle;

import android.support.v7.widget.LinearLayoutManager;

import android.support.v7.widget.RecyclerView;

import android.util.Log;

import android.view.View;

import android.widget.Button;

import android.widget.RadioButton;

import android.widget.RadioGroup;

import android.widget.Toast;

import com.android.volley.Request;

import com.android.volley.RequestQueue;

import com.android.volley.Response;

import com.android.volley.VolleyError;

import com.android.volley.toolbox.StringRequest;

import com.android.volley.toolbox.Volley;

import com.kgfsl.ecampusstudent.Activity.ClassListActivity;

import com.kgfsl.ecampusstudent.Activity.ClassModel;

import org.json.JSONArray;

import org.json.JSONException;

import org.json.JSONObject;

import java.util.ArrayList;

import java.util.HashMap;

import java.util.Map;

public class AttendanceActivity extends AppCompatActivity {

RecyclerView recyclerView;

LinearLayoutManager linearLayoutManager;

String id;

URL url;

AttendanceListAdapter attendanceListAdapter;

Button submit;

ArrayList<AttendanceModel> data =new ArrayList<>();

RadioGroup radioGroup;

AttendanceModel attendanceModel;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_attendance);

submit=(Button)findViewById(R.id.submit);

id = getIntent().getStringExtra("id");

recyclerView=(RecyclerView)findViewById(R.id.recyclerview);

linearLayoutManager=new LinearLayoutManager(getApplicationContext(),LinearLayoutManager.VERTICAL,false);

recyclerView.setLayoutManager(linearLayoutManager);

radioGroup = (RadioGroup) findViewById(R.id.radio);

getStudentList(id);

submit.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

AlertDialog.Builder builder = new AlertDialog.Builder(AttendanceActivity.this);

builder.setTitle("Attendance Register");

builder.setMessage("Are You sure want to Register Attendance!");

builder.setPositiveButton("Yes", new DialogInterface.OnClickListener() {

@Override

public void onClick(DialogInterface dialog, int which) {

calculateAttendance();

}

});

builder.setNegativeButton("No", new DialogInterface.OnClickListener() {

@Override

public void onClick(DialogInterface dialog, int which) {

dialog.dismiss();

}

});

AlertDialog alertDialog = builder.create();

alertDialog.show();

}

});

}

private void calculateAttendance(){

JSONArray jsonArray = new JSONArray();

for(int i=0;i<data.size();i++){

try {

JSONObject dataObj = new JSONObject();

int id = data.get(i).getId();

dataObj.put("sid", data.get(i).getId());

if (data.get(i).getPos() == true) {

dataObj.put("status", "P");

} else if (data.get(i).getPos() == false) {

dataObj.put("status", "A");

}

jsonArray.put(dataObj);

}catch (JSONException e) {

e.printStackTrace();

}

}

JSONObject jsonObject = new JSONObject();

try {

jsonObject.put("data",jsonArray);

jsonObject.put("cid",id);

int selectedId = radioGroup.getCheckedRadioButtonId();

RadioButton radioButton = (RadioButton) findViewById(selectedId);

jsonObject.put("hour",radioButton.getText());

pushAttendance(String.valueOf(jsonObject));

} catch (JSONException e) {

e.printStackTrace();

}

}

private void pushAttendance(final String data) {

RequestQueue requestQueue = Volley.newRequestQueue(this);

StringRequest stringRequest = new StringRequest(Request.Method.POST, url.attendanceregister, new Response.Listener<String>() {

@Override

public void onResponse(String response) {

toast(response);

//getStudentList(id);

}

}, new Response.ErrorListener() {

@Override

public void onErrorResponse(VolleyError error) {

toast(String.valueOf(error));

}

}){

@Override

protected Map<String, String> getParams()

{

Map<String, String> params = new HashMap<String, String>();

params.put("data", data);

return params;

}

};

requestQueue.add(stringRequest);

}

private void getStudentList(final String id) {

data=new ArrayList<>();

RequestQueue requestQueue = Volley.newRequestQueue(this);

StringRequest stringRequest = new StringRequest(Request.Method.POST, url.studentlist, new Response.Listener<String>() {

@Override

public void onResponse(String response) {

try {

JSONObject jsonObject = new JSONObject(response);

JSONArray jsonArray = jsonObject.getJSONArray("data");

for(int i=0;i<jsonArray.length();i++){

JSONObject jsonObject1 = jsonArray.getJSONObject(i);

int id = jsonObject1.getInt("id");

String name = jsonObject1.getString("name");

data.add(new AttendanceModel(id,name));

}

attendanceListAdapter = new AttendanceListAdapter(AttendanceActivity.this,data,id);

recyclerView.setAdapter(attendanceListAdapter);

} catch (JSONException e) {

toast(String.valueOf(e));

e.printStackTrace();

}

}

}, new Response.ErrorListener() {

@Override

public void onErrorResponse(VolleyError error) {

toast(String.valueOf(error));

}

}){

@Override

protected Map<String, String> getParams()

{

Map<String, String> params = new HashMap<String, String>();

params.put("id", id);

return params;

}

};

requestQueue.add(stringRequest);

}

private void toast(String s) {

Toast.makeText(getApplicationContext(),s,Toast.LENGTH\_SHORT).show();

}

}package com.kgfsl.ecampusstudent;

import android.content.Context;

import android.content.Intent;

import android.support.v7.widget.RecyclerView;

import android.view.LayoutInflater;

import android.view.View;

import android.view.ViewGroup;

import android.widget.CompoundButton;

import android.widget.LinearLayout;

import android.widget.Switch;

import android.widget.TextView;

import android.widget.Toast;

import com.kgfsl.ecampusstudent.Activity.MainActivity;

import java.util.ArrayList;

/\*\*

\* Created by Madhan on 25-12-2017.

\*/

public class AttendanceListAdapter extends RecyclerView.Adapter<AttendanceListAdapter.Holder> {

Context ctx;

ArrayList<AttendanceModel> datas;

String cid;

public AttendanceListAdapter(Context context, ArrayList<AttendanceModel> data ,String cid){

this.ctx=context;

this.datas=data;

this.cid=cid;

}

@Override

public AttendanceListAdapter.Holder onCreateViewHolder(ViewGroup parent, int viewType) {

return new AttendanceListAdapter.Holder(LayoutInflater.from(parent.getContext()).inflate(R.layout.row\_attendance,parent,false));

}

@Override

public void onBindViewHolder(AttendanceListAdapter.Holder holder, final int position) {

holder.name.setText(datas.get(position).getName());

holder.switchb.setOnCheckedChangeListener(new CompoundButton.OnCheckedChangeListener() {

public void onCheckedChanged(CompoundButton buttonView, boolean isChecked) {

if(isChecked) {

datas.get(position).setPos(true);

}else{

datas.get(position).setPos(false);

}

}

});

holder.row.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

Intent intent = new Intent(ctx, MainActivity.class);

intent.putExtra("id",String.valueOf(datas.get(position).getId()));

intent.putExtra("cid",String.valueOf(cid));

intent.putExtra("FLAGG","ATT");

ctx.startActivity(intent);

}

});

holder.switchb.setChecked(datas.get(position).getPos());

}

private void toast(String s) {

Toast.makeText(ctx,s,Toast.LENGTH\_SHORT).show();

}

@Override

public int getItemCount() {

return datas.size();

}

public class Holder extends RecyclerView.ViewHolder {

TextView name;

Switch switchb;

LinearLayout row;

public Holder(View itemView) {

super(itemView);

name=(TextView)itemView.findViewById(R.id.name);

switchb =(Switch)itemView.findViewById(R.id.switchb);

row =(LinearLayout) itemView.findViewById(R.id.row);

}

}

}