|  |  |
| --- | --- |
| **Prepared by:** | HOPE Development Team |
| **Version:** | 1.1 |
| **Date:** | 05/29/2014 |
|  |  |

HOPE System

Detailed Design Document

[REVISION HISTORY 3](#_Toc387911378)

[1 Introduction 4](#_Toc387911379)

[1.1 Purpose 4](#_Toc387911380)

[1.2 Scope 4](#_Toc387911381)

[1.2.1 In Scope 4](#_Toc387911382)

[1.2.2 Out of Scope 4](#_Toc387911383)

[2 System Overview 5](#_Toc387911384)

[3 Environments and Conventions 6](#_Toc387911385)

[3.1 Production Environment 6](#_Toc387911386)

[3.2 Development Environment 6](#_Toc387911387)

[3.3 Test Environment 6](#_Toc387911388)

[3.4 Naming and Coding Conventions 6](#_Toc387911389)

[3.4.1 Java Naming Conventions 6](#_Toc387911390)

[3.4.2 Other Naming Conventions 7](#_Toc387911391)

[4 Source Code Hierarchy 8](#_Toc387911392)

[5 Web Pages 9](#_Toc387911393)

[5.1 Test Bank 9](#_Toc387911394)

[5.2 Login 9](#_Toc387911395)

[5.3 Teachers 9](#_Toc387911396)

[5.3.1 XXX 9](#_Toc387911397)

[5.3.2 YYY 9](#_Toc387911398)

[5.3.3 ZZZ 9](#_Toc387911399)

[5.3.4 AAA 9](#_Toc387911400)

[5.4 Students 9](#_Toc387911401)

[5.4.1 AAA 9](#_Toc387911402)

[5.4.2 BBB 9](#_Toc387911403)

[5.5 Parents 10](#_Toc387911404)

[5.6 Teacher in Charge 10](#_Toc387911405)

[5.7 Administrator 10](#_Toc387911406)

[6 Services 11](#_Toc387911407)

[6.1 AAA 11](#_Toc387911408)

[6.2 BBB 11](#_Toc387911409)

[6.3 CCC 11](#_Toc387911410)

[6.4 DDD 11](#_Toc387911411)

[7 Appendix A: Definitions, Acronyms and Abbreviations 12](#_Toc387911412)

[Table 1 Java naming conventions 7](#_Toc387911413)

[Table 2 Acronyms and abbreviations 12](#_Toc387911414)

[Figure 1 HOPE Architecture Diagram 5](file:///C:\Users\z419731\Documents\My%20Projects\LanKing\HOPE_Design.docx#_Toc387911415)

# REVISION HISTORY

This section specifies changes made to the document after it has been inspected and signed off. This section ensures that development and/or user requests and changes have traceability.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Date** | **Name** | **Source** | Description of Change | **Version Number** |
| 5/15/2014 | Larry Zhang |  | Original Document | 1.0 |
| 5/28/2014 | Ning Chen |  | Adding Android related Document | 1.1 |

# Introduction

## Purpose

## Scope

### In Scope

### Out of Scope

# System Overview



JSP

DB Manager

Hope DB

**Presentation Layer**

**Business Logic Layer**

**Resource Layer**

CSS

JavaScript

HTML

Common

Service (event, notification)

Servlet

File Helper

DB Helper

RESTFul



Brower

Android App

Data

**HOPE Web Server**

Figure HOPE Architecture Diagram

# Environments and Conventions

Figure 2 Android System Diagram



Restful Service client(retrofit)

Activity

**Communication layer**

**Service Layer**

**Presentation layer**

Server request queue

(tape)

Event Service bus

(otto)

Business logic

Javascript

Webview



Web Server

Android App

Data service layer

Ormlite/Sqlite

**Android Client**

## Production Environment

Hardware: Intel

OS: Linux

Database: MySQL

Programming language: J2EE

Android Hardware: Dell Venue 8

OS : Android 4.3 ( Jelly Bean) ,possible 4.4(Kitkat) in the future

Android client DB : sqlite3

Development tool : JDK1.6 ,Android studio (0.58) with Gradle 1.10

## Development Environment

## Test Environment

## Naming and Coding Conventions

### Java Naming Conventions

The Java naming convention recommended by SUN should be followed. Here are some of the key rules:

|  |  |  |
| --- | --- | --- |
| **Identifier Type** | **Rules for Naming** | **Examples** |
| Packages | The prefix of a unique package name is always written in all-lowercase ASCII letters and should be one of the top-level domain names, currently com, edu, gov, mil, net, org, or one of the English two-letter codes identifying countries as specified in ISO Standard 3166, 1981.  Subsequent components of the package name vary according to an organization's own internal naming conventions. Such conventions might specify that certain directory name components be division, department, project, machine, or login names. | com.sun.eng  com.lanking.hope  com.lanking.hope.servlet |
| Classes | Class names should be nouns, in mixed case with the first letter of each internal word capitalized. Try to keep your class names simple and descriptive. Use whole words-avoid acronyms and abbreviations (unless the abbreviation is much more widely used than the long form, such as URL or HTML). | class String; class StringBuilder; |
| Interfaces | Interface names should be capitalized like class names. | interface Runnable; interface CharSequence; |
| Methods | Methods should be verbs, in mixed case with the first letter lowercase, with the first letter of each internal word capitalized. | run(); runFast(); getBackground(); |
| Variables | Variables should be in mixed case with a lowercase first letter. Internal words start with capital letters. Variable names should not start with underscore \_ or dollar sign $ characters, even though both are allowed.  Variable names should be short yet meaningful. The choice of a variable name should be mnemonic- that is, designed to indicate to the casual observer the intent of its use. One-character variable names should be avoided except for temporary "throwaway" variables. Common names for temporary variables are i, j, k, m, and n for integers; c, d, and e for characters. | int i;  char c;  float myWidth; |
| Constants | The names of variables declared class constants and of ANSI constants should be all uppercase with words separated by underscores ("\_"). (ANSI constants should be avoided, for ease of debugging.) | static final int MIN\_WIDTH = 4;  static final int MAX\_WIDTH = 999;  static final int GET\_THE\_CPU = 1; |

Table 1 Java naming conventions

### Other Naming Conventions

* Directory names should be in all-lowercase ASCII letters
* Except for Java classes, file names should in mixed case with a lowercase first letter. This rule applies to jsp files, javascript files, css files, etc.
* JavaScript naming conventions should be similar to the Java naming conventions

# Source Code Hierarchy

The source code directories are as follows:

* hope (Eclipse project name)
  + dist (war file)
  + doc (documentation)
  + src (Java source code)
    - com.lanking.hope.common (common classes shared by Servlet and RESTful)
    - com.lanking.hope.data (data classes)
    - com.lanking.hope.db (DB helper classes)
    - com.lanking.hope.rest (RESTful classes)
    - com.lanking.hope.service (services, usually run in new threads)
    - com.lanking.hope.servlet (Servlets)
    - com.lanking.hope.util (utility classes)
  + web (web pages including JSP, HTML, JavaScript, CSS and images)
    - images
    - includes
      * js
      * css
    - WEB-INF
      * lib
      * classes
    - Other directories

# 

# Services

**5.1 Question Pool Functions**

public List<Tap\_Point> listTapPoints(String catalog) throws SQLException;

public List<Tap\_Point> listTapPoints() throws SQLException;

public List<Grade> listGrades() throws SQLException;

public List<Catalog> listCatalogs() throws SQLException;

public List<Q\_Type> listQTypes() throws SQLException;

public Question getQuestion(String userId, int qId) throws SQLException;

public int copyQuestion(String userId, int qId) throws SQLException;

public int addQuestion(String userId, Question question) throws SQLException;

public int submitQuestion(String userId, int qId) throws SQLException;

public int rejectQuestion(String userId, int qId) throws SQLException;

public int acceptQuestion(String userId, int qId) throws SQLException;

public int publishQuestion(String userId, int qId) throws SQLException;

public int unpublishQuestion(String userId, int qId) throws SQLException;

public int dropQuestion(String userId, int qId) throws SQLException;

public Q\_Group getQGroup(String userId, int groupId) throws SQLException;

public int addQGroup(String userId, Q\_Group qGroup) throws SQLException;

public int dropQGroup(String userId, int groupId) throws SQLException;

public List<String> findQGroups(String userId, String name) throws SQLException;

public List<String> findQuestions(String userId, Q\_Search search) throws SQLException;

public int addSchool(String userId, School school) throws SQLException;

public List<School> listSchools() throws SQLException;

public List<S\_Class> listSClasses(String school) throws SQLException;

public List<S\_Class> listSClasses() throws SQLException;

public int addSClass(String userId, S\_Class sClass) throws SQLException;

**5.2 Account Management Functions**

public String addInspector(String userId, Inspector inspector) throws SQLException;

public String addBuilder(String userId, Builder builder) throws SQLException;

public String addAdmin(String userId, Admin admin) throws SQLException;

public String addGuardian(String userId, Guardian guardian) throws SQLException;

public String addStudent(String userId, Student student) throws SQLException;

public String addTeacher(String userId, Teacher teacher) throws SQLException;

public String addLocalAdmin(String userId, LocalAdmin localAdmin) throws SQLException;

5.3 Teacher's Planning Functions

public int addQSet(String userId, Q\_Set qSet) throws SQLException;

public List<Q\_Set> findMyQSets(String userId, String type, String name) throws SQLException;

public int dropMyQSet(String userId, int qSetId) throws SQLException;

public Q\_Set getMyQSet(String userId, int qSetId) throws SQLException;

public int addQSetToMyTopic(String userId, int topicId, int qSetId) throws SQLException;

public int dropQSetFromMyTopic(String userId, int topicId, int qSetId) throws SQLException;

public List<Q\_Set> findQSetsInMyTopic(String userId, int topicId) throws SQLException;

public int addPlan(String userId, Plan plan) throws SQLException;

public List<Plan> listMyPlans(String userId) throws SQLException;

public int dropMyPlan(String userId, int planId) throws SQLException;

public int getMyPlan(String userId, int planId) throws SQLException;

public int addTopicToMyPlan(String userId, int planId, Topic topic) throws SQLException;

public int dropTopicFromMyPlan(String userId, int planId, int topicId) throws SQLException;

public List<Topic> listTopicsInMyPlan(String userId, int planId) throws SQLException;

public Topic getMyTopic(String userId, int topicId) throws SQLException;

# Appendix A: Definitions, Acronyms and Abbreviations

|  |  |
| --- | --- |
| **Acronym** | **Definition** |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

Table 2 Acronyms and abbreviations