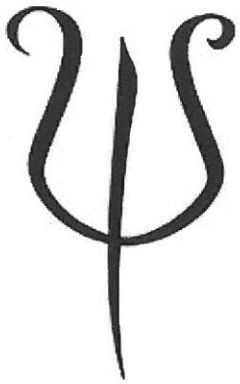
Index



Team Psi

Member:	E-mail:	Student ID:	Phone Number:
Thomas Hoang	thomas.hoang09@gmail.com	1000825196	(817)584-7174
Linh Vu	linhnvu@mavs.uta.edu	1000638144	(682)521-3233
Dennis Duong	dennis.duong@mavs.uta.edu	1000335195	(972)742-1065
JR Ladera	emmanueljrladera@gmail.com	1000817843	(469)877-1939

Project: UTA Schedule Planner

Team Name: Psi

Class: CSE 3310; Spring 2015

Module: System Requirements Analysis (SRA)

Deliverable: SRA Document

Version: 1.1 Date: 5/7/2015

Contributors:

Thomas Hoang

Linh Vu

Dennis Duong

JR Ladera

Revision History

Version Number	Date	Originator	Reason for Change	High level description of changes
1.0	3/31/2015	Thomas Hoang, Linh Vu, Dennis Duong, JR Ladera	Initial Draft	
1.1	5/6/2015	Thomas Hoang	Revision for final product	

Table of Contents

1. INTRODUCTION & PROJECT OVERVIEW

- **Tasks:** Design, implement, and test an application for android devices, where the user may:
  - **i.** Create User(s)
  - **ii.** Create Time Restrictions
  - **iii.** Create Schedules
  - **iv.** View Schedules
  - **v.** Add or Remove classes to their schedule
  - **vi.** Manage their UTA school schedule(s).
- **Users:** Android UTA students
- **Development Model:** Incremental Development

2. OBJECTIVES

2.1 BUSINESS Objectives

Objective 1: User Creation: All users must create an account and provide:

- A unique username
- A password

Objective 2: *Login Functionality*: All users must login with a username and password which was created during the *User Creation* process.

Objective 3: *Time Restrictions Functionality*: The current user may create time periods where the application hides unwanted classes.

Objective 4: *Create Schedule Functionality*: The current user chooses classes to add to their schedule.

Objective 5: *Manage Schedule Functionality*: The current user may add additional classes, edit time restrictions, add additional schedules or remove a schedule altogether.

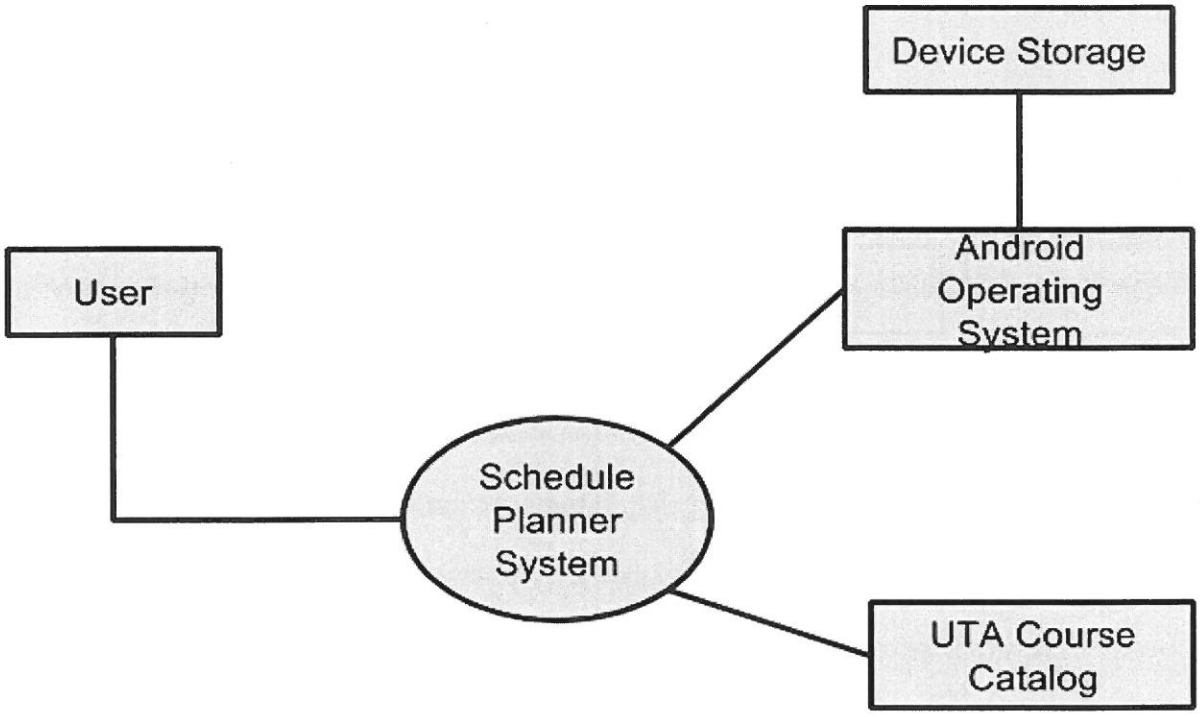
2.2 SYSTEM Objectives

Objective 1: System will be Android-based.

Objective 2: UTA class catalog shall be used as a database to supplement Create Schedule Functionality.

Objective 3: Schedules shall be saved to the device used.

3. Project Context Diagram



4. Systems Requirements

Requirement Title: (*required)	Log in
Sequence No: (*required)	1.0
Short description: (*required)	Log in screen interface, user logs in
Detailed Description: (*required)	1. Shall read username and password from user input. 2. App shall check username and password for validating. 3. App shall save usernames and passwords locally on device.
Pre-Conditions: (optional)	
Post Conditions: (optional)	Main Interface Main log in screen if failed log in
Other attributes: (optional)	Username: minimum 4 characters, max 12, alphanumeric only Password: minimum 8 characters, maximum 15, alphanumeric only

Requirement Title: (*required)	Create User Page
Sequence No: (*required)	1.1
Short description: (*required)	Create a new user
Detailed Description: (*required)	1. App shall validate username, password, and verify password.
Pre-Conditions: (optional)	
Other attributes: (optional)	Username/password are stored on device Username: minimum 4 characters, max 12, alphanumeric only Password: minimum 8 characters, maximum 15, alphanumeric only

Requirement Title: (*required)	Time restrictions
Sequence No: (*required)	2.3
Short description: (*required)	User inputs time restrictions
Detailed Description: (*required) Notify user to push button to enter time with text	1. App shall give the user option to set up time restriction(s). 2. App shall provide the user a means of entering information.
Pre-Conditions: (optional)	User is logged in Internet connection available
Post Conditions: (optional)	Return to last page viewed
Other attributes: (optional)	Maximum of 10 time restrictions Days available are Monday, Tuesday, Wednesday, Thursday, Friday, Saturday Times available is 12:00 a.m. to 11:59 p.m.

Requirement Title: (*required)	User Settings
Sequence No: (*required)	2.4
Short description: (*required)	Change user password, remove the user
Detailed Description: (*required)	1. App shall allow user to change their password. 2. App shall allow user to remove a user.
Pre-Conditions: (optional)	User must be logged in
Post Conditions: (optional)	
Other attributes: (optional)	Username/password are stored on device Username: minimum 4 characters, max 12, alphanumeric only Password: minimum 8 characters, maximum 15, alphanumeric only

Requirement Title: (*required)	Create a schedule
Sequence No: (*required)	2.1
Short description: (*required)	User input informations to creating schedule
Detailed Description: (*required)	1. App shall check for saved time restriction. 2. App shall allow user to enter course information. 3. App shall connect to UTA catalog to create a list of schedules. 4. App shall check for schedules conflicts. 5. App shall account for Saturday classes. 6. App shall save schedules locally to device.
Pre-Conditions: (optional)	Internet connection must be available Storage space must be available
Post Conditions: (optional)	
Other attributes: (optional)	Entries for course subjects must be letters (A-Z), upper/lower case Entries for course numbers must be numbers (0-9) Days available are Monday, Tuesday, Wednesday, Thursday, Friday, Saturday Times available is 12:00 a.m. to 11:59 p.m. Up to 10 schedules

Requirement Title: (*required)	Manage My Schedules
Sequence No: (*required)	2.2
Short description: (*required)	User manages different lists of schedules that they have created
Detailed Description: (*required)	1. App shall show saved schedules 2. App shall allow user to modify saved schedules. 3. App shall save up to 10 schedules.
Pre-Conditions: (optional)	
Post Conditions: (optional)	
Other attributes: (optional)	Modify: add class, remove class, create schedule, remove schedule, edit time restrictions

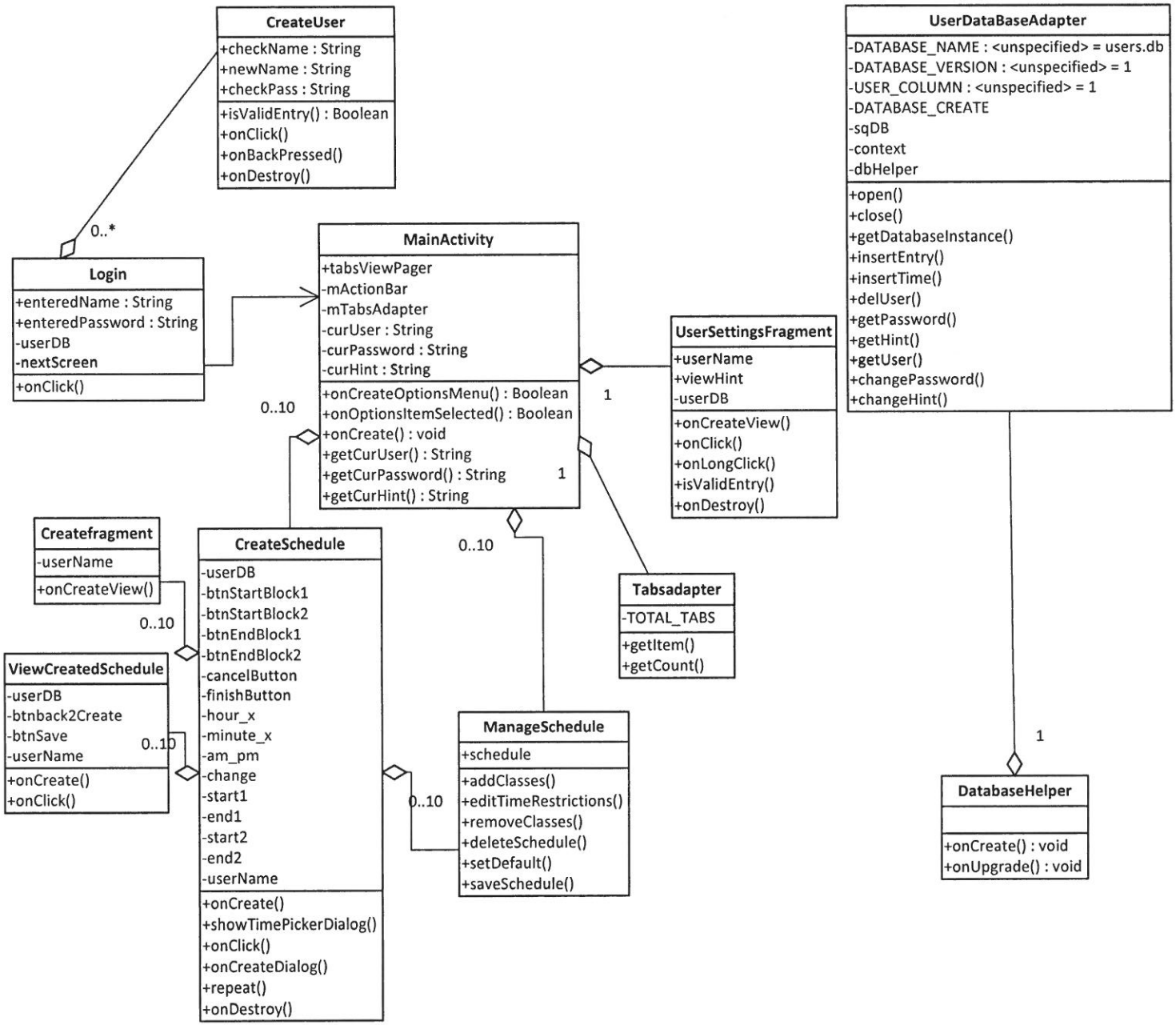
5. Software Processes and Infrastructure

5.1 Hardware and Infrastructure

- Android device (Operating System 2.3 or higher)

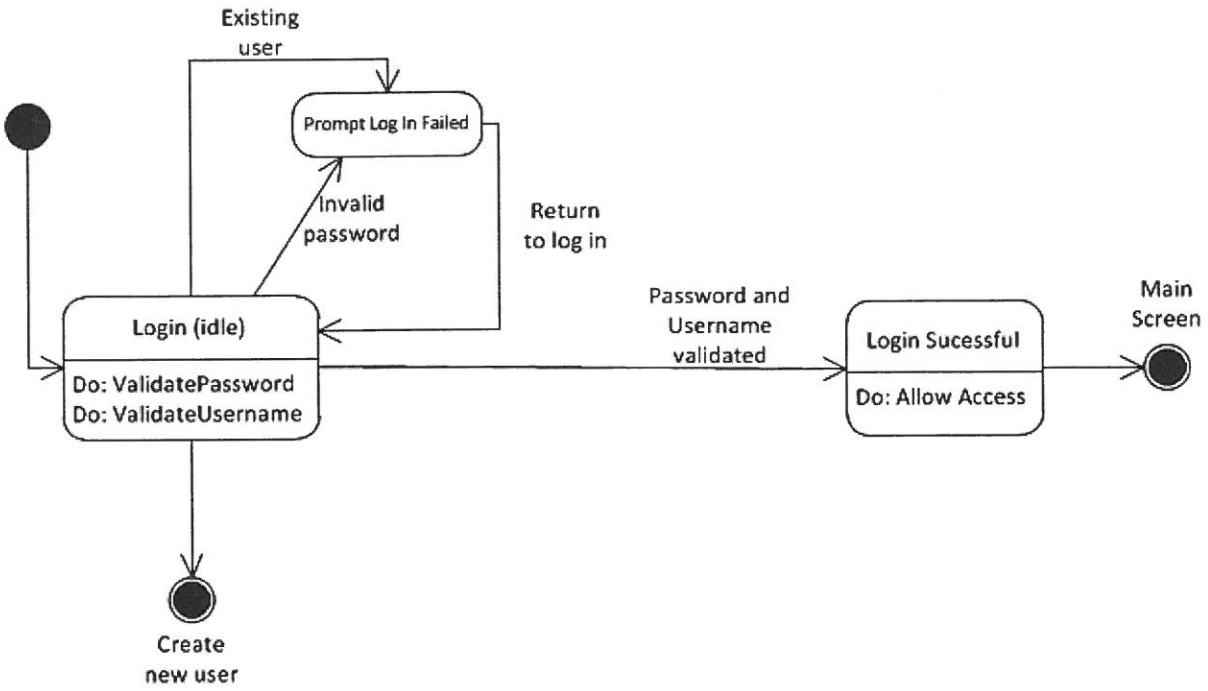
5.2 UML Diagrams

ON NEXT PAGE

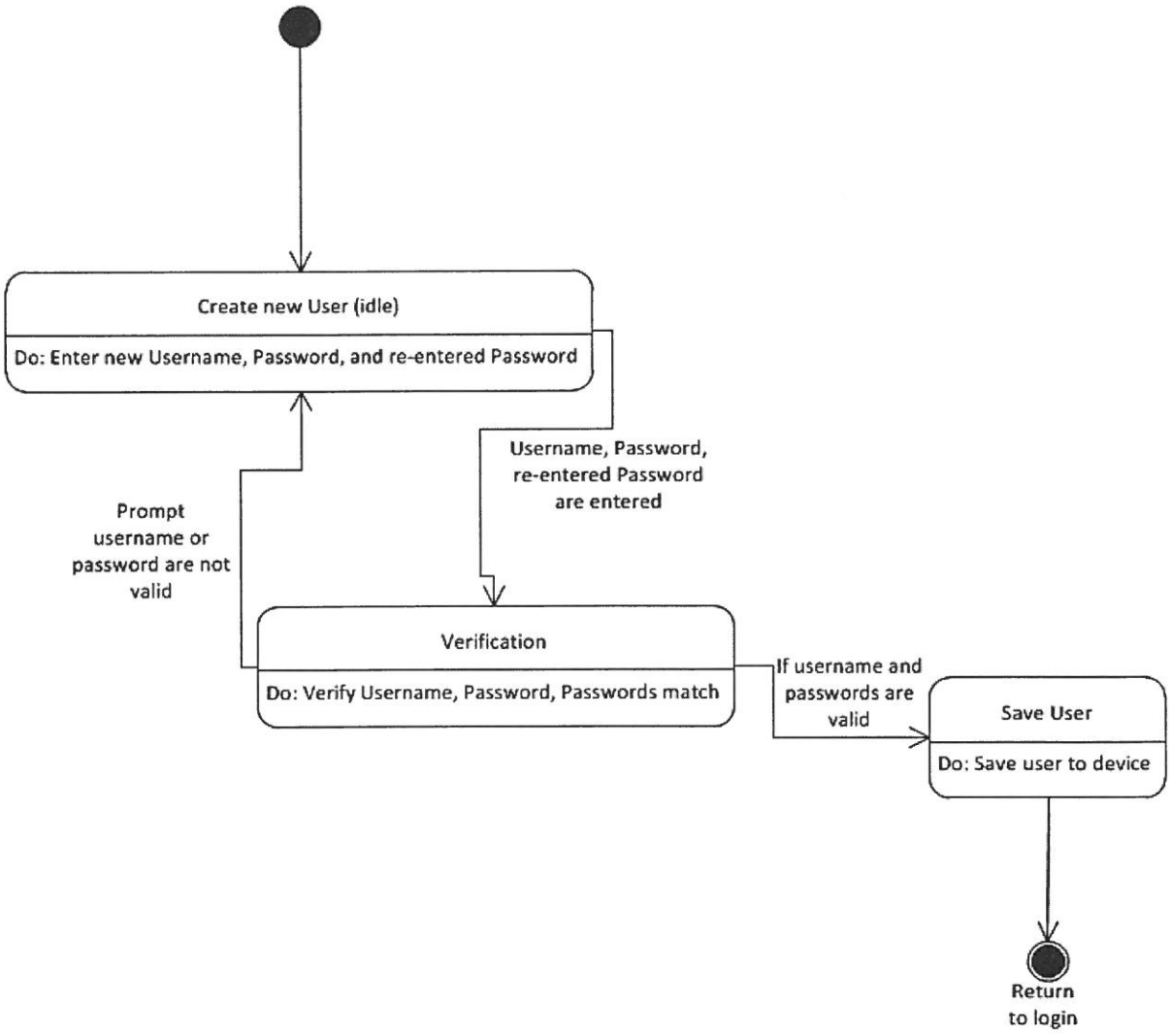

CreateUser
+checkName : String
+newName : String
+checkPass : String
+isValidEntry() : Boolean
+onClick()
+onBackPressed()
+onDestroy()
0..*
Login
+enteredName : String
+enteredPassword : String
-userDB
-nextScreen
+onClick()
MainActivity
+tabsViewPager
-mActionBar
-mTabsAdapter
-curUser : String
-curPassword : String
-curHint : String
+onCreateOptionsMenu() : Boolean
+onOptionsItemSelected() : Boolean
+onCreate() : void
+getCurUser() : String
+getCurPassword() : String
+getCurHint() : String
0..10
1
1
UserSettingsFragment
+userName
+viewHint
-userDB
+onCreateView()
+onClick()
+onLongClick()
+isValidEntry()
+onDestroy()
UserDataBaseAdapter
-DATABASE_NAME : <unspecified> = users.db
-DATABASE_VERSION : <unspecified> = 1
-USER_COLUMN : <unspecified> = 1
-DATABASE_CREATE
-sqDB
-context
-dbHelper
+open()
+close()
+getDatabaseInstance()
+insertEntry()
+insertTime()
+delUser()
+getPassword()
+getHint()
+getUser()
+changePassword()
+changeHint()
Createfragment
-userName
+onCreateView()
0..10
CreateSchedule
-userDB
-btnStartBlock1
-btnStartBlock2
-btnEndBlock1
-btnEndBlock2
-cancelButton
-finishButton
-hour_x
-minute_x
-am_pm
-change
-start1
-end1
-start2
-end2
-userName
+onCreate()
+showTimePickerDialog()
+onClick()
+onCreateDialog()
+repeat()
+onDestroy()
0..10
Tabsadapter
-TOTAL_TABS
+getItem()
+getCount()
ViewCreatedSchedule
-userDB
-btnback2Create
-btnSave
-userName
+onCreate()
+onClick()
0..10
ManageSchedule
+schedule
+addClasses()
+editTimeRestrictions()
+removeClasses()
+deleteSchedule()
+setDefault()
+saveSchedule()
0..10
1
DatabaseHelper
+onCreate() : void
+onUpgrade() : void

State Diagrams

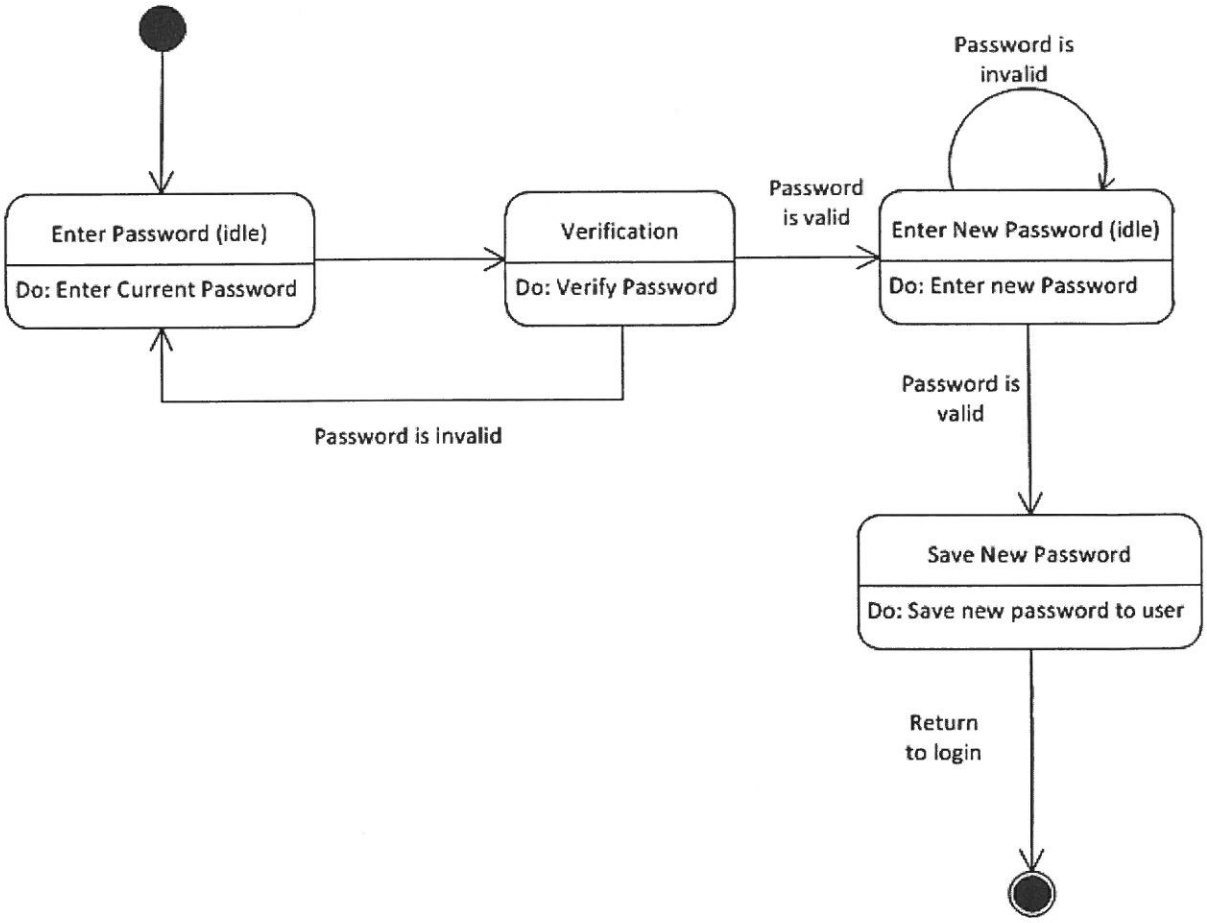
Log In Diagram



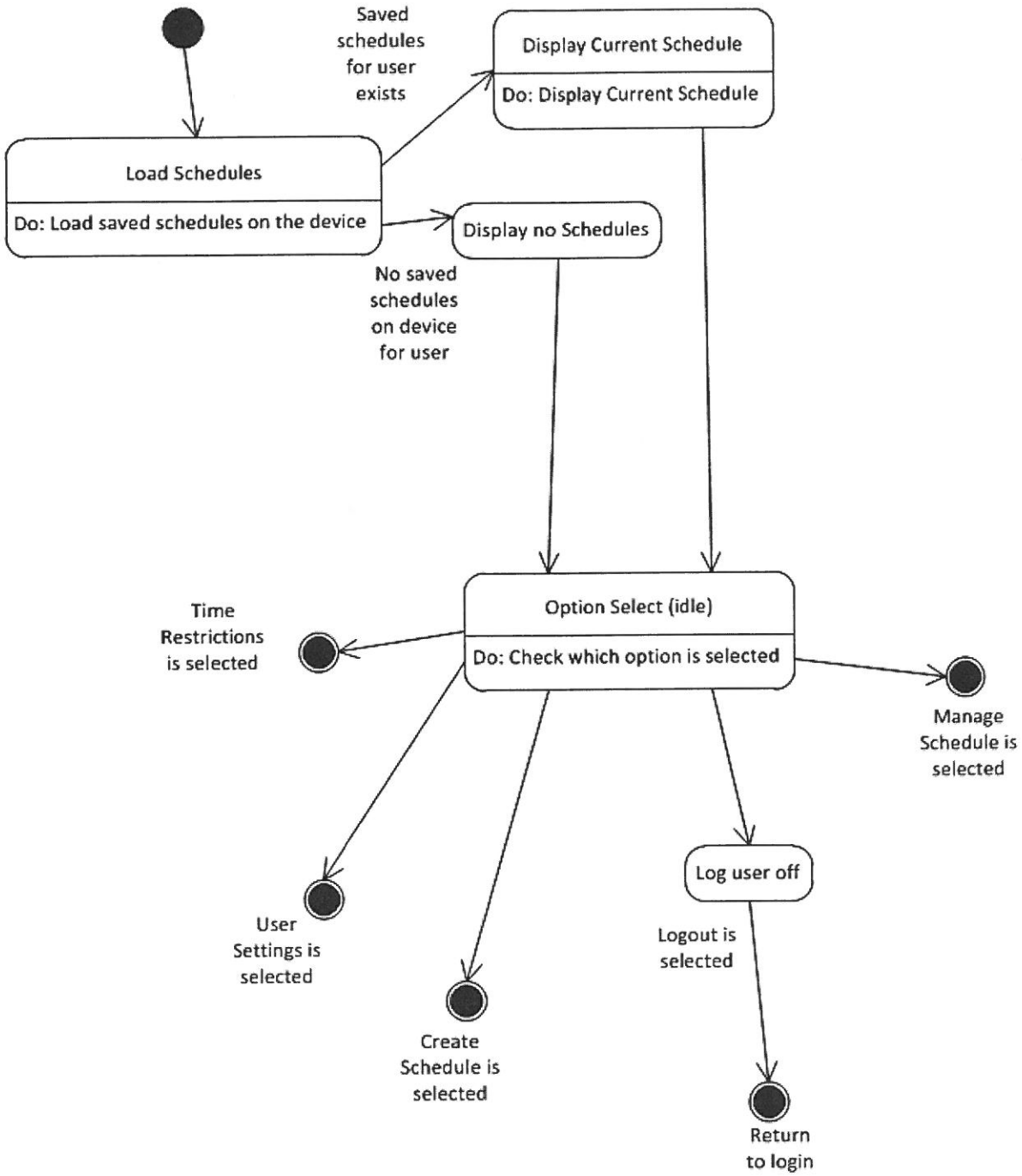
Create New User Diagram



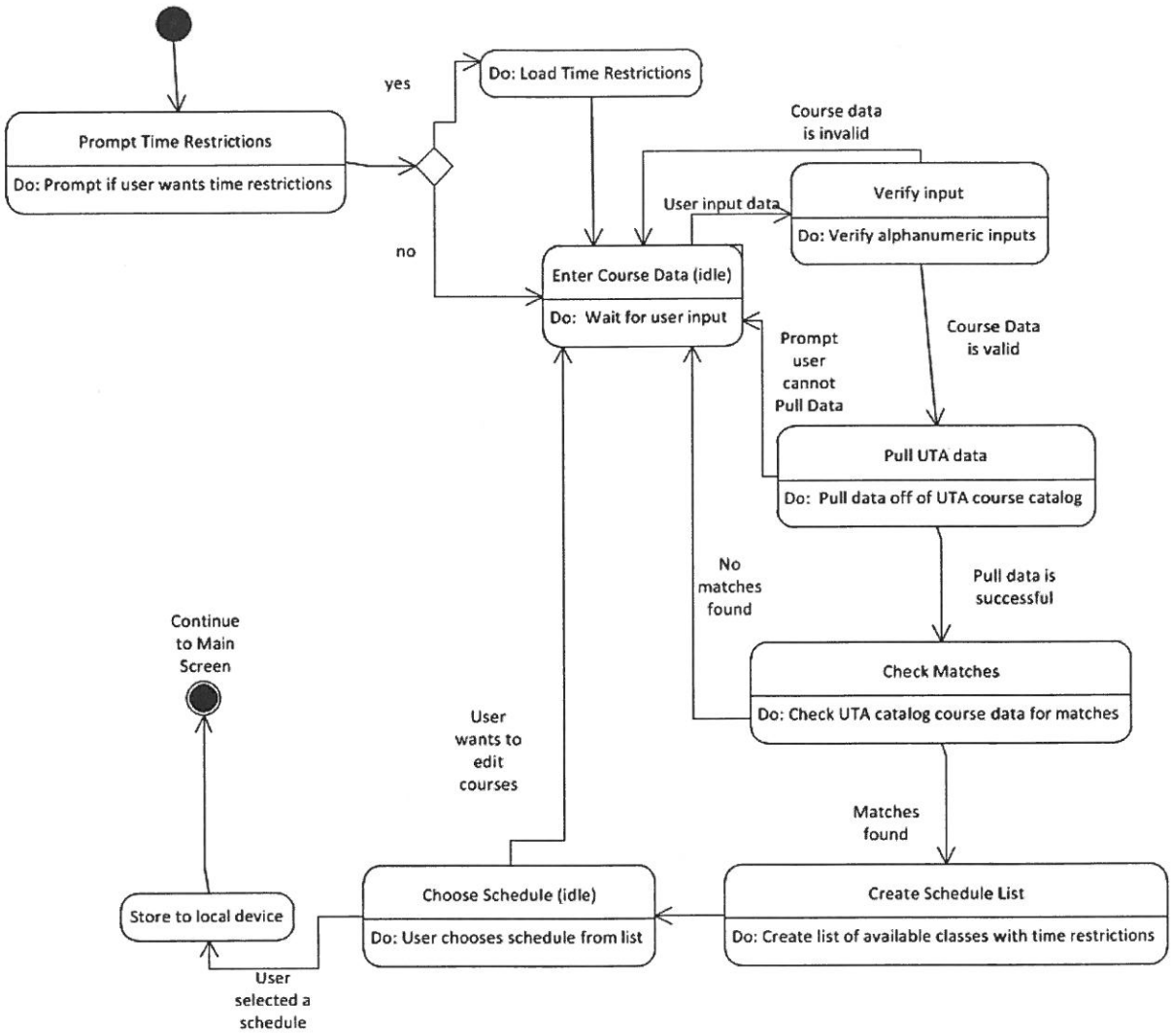
User Settings Diagram



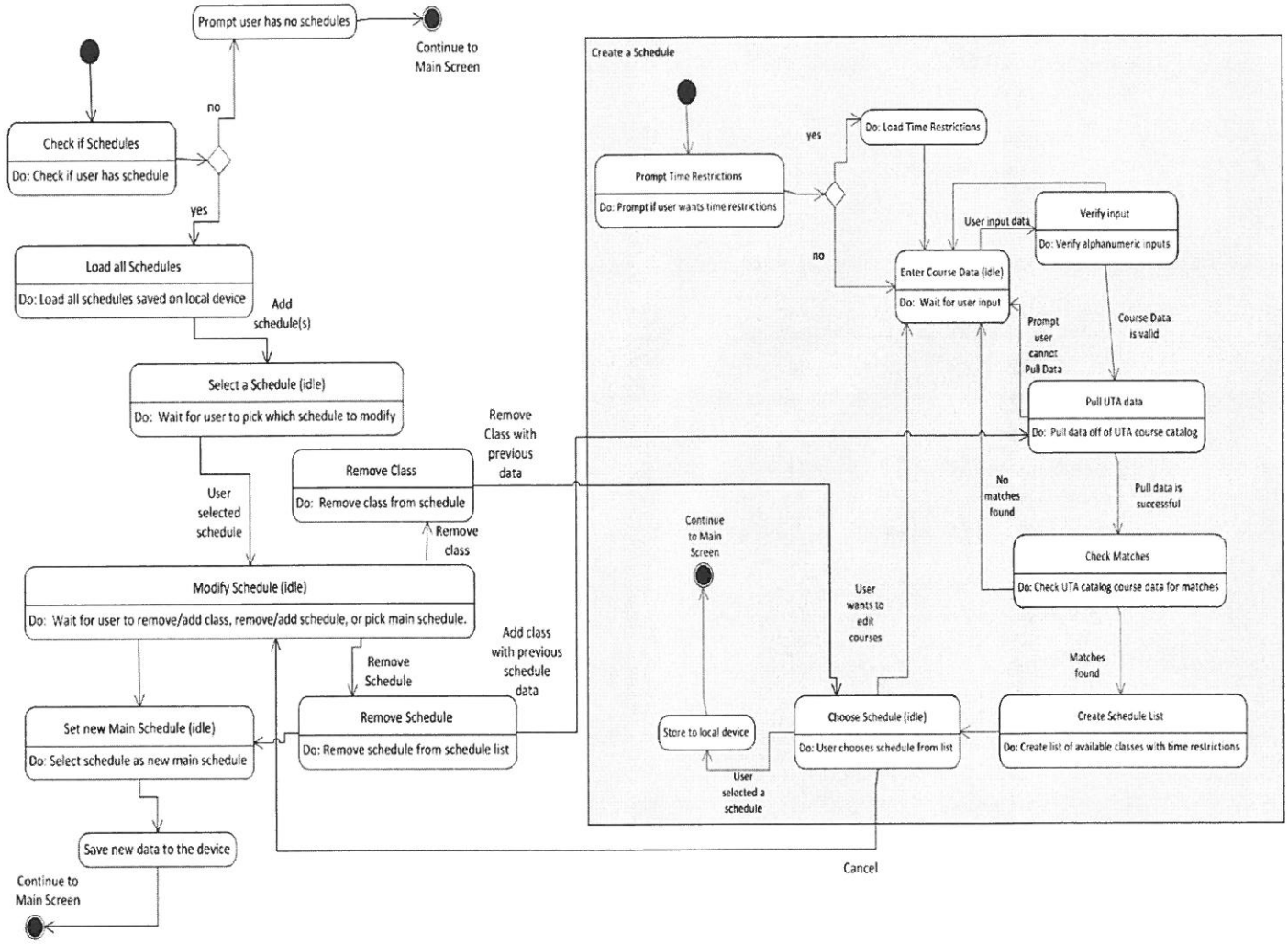
Main Screen Diagram



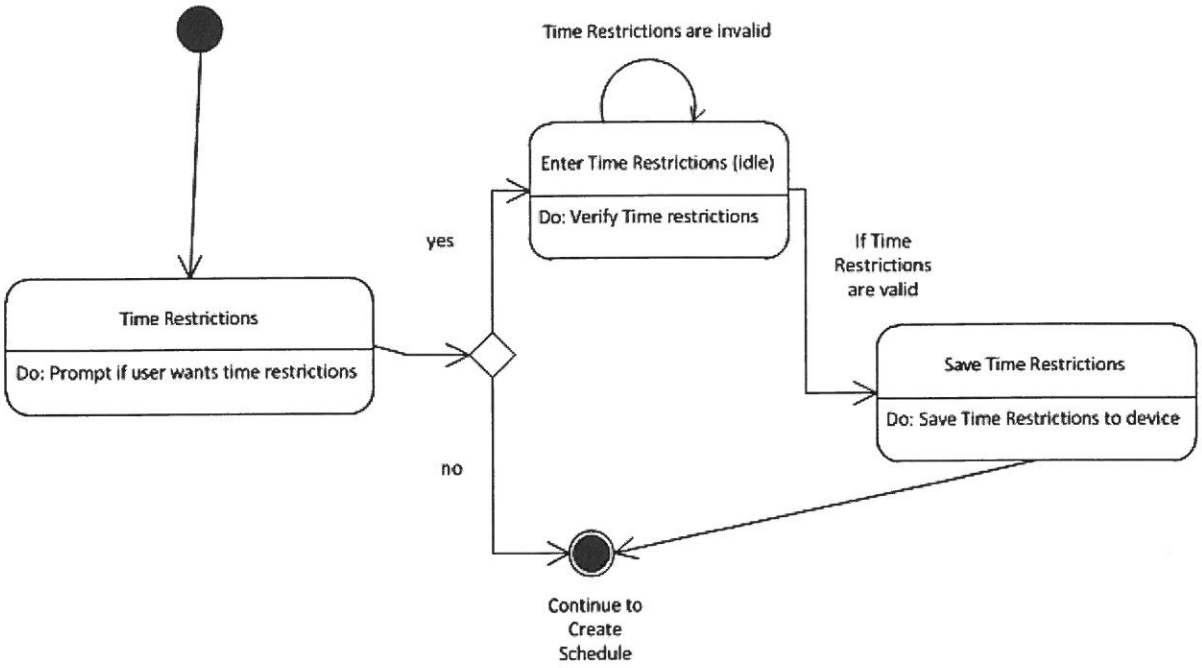
Create a Schedule Diagram



Manage Schedules Diagram



Time Restrictions Diagram



5.3 Conceptual Data Model-Database

- UTA course catalog

5.4 Screenshots

- None available at this time

5.5 Test Plan

Test Cases: "Log in"

Project Name: **UTA Schedule Planner**
Test Case Name: **Login**
Test Case Id: **CSE3310/Spring 2015/Team Psi/ Login**

Test Case No.	Test Case Description	Expected results	Outcome Pass, Fail, Other (comments)
TC1	Enter a valid username that exists and valid Password for that username. Valid usernames and passwords consists of alphanumeric characters only and the username must be existing in the database.	System should log entered username in.	Pass
TC2	Enter a valid username and invalid password.Valid usernames and passwords consists of alphanumeric characters only and the username must be existing in the database.	System should prevent user from entry with a prompt to notify user.	Pass

TC3	Enter an invalid username and valid password. Valid usernames and passwords consists of alphanumeric characters only and the username must be existing in the database.	System should prevent user from entry with a prompt to notify user.	**Pass**
TC4	Enter an invalid username and invalid password. Invalid usernames and passwords consists of characters other than alphanumeric characters.	System should prevent user from entry with a prompt to notify user.	**Pass**
TC5	Tab Create New User button.	System should direct user to create user screen.	**Pass**
TC6	Internet connection is available	System should run normally with no notification to the user.	**Inconclusive**
TC7	Internet connection is not available	System should notify user that there is no connection available.	**Inconclusive**

Test Cases: "Create New User"

Project Name: UTA Schedule Planner
Test Case Name: Create New User
Test Case Id: CSE3310/Spring 2015/Team Psi/ Create_New_User

Test Case No.	Test Case Description	Expected results	Outcome Pass, Fail, Other (comments)
TC1	Tab into the Username and password fields and enter a valid Username/Password. Valid usernames and passwords consists of alphanumeric characters only and the username must not be already existing in the database.	System should create a new user.	Pass
TC2	Tab into the Username and Password fields and enter an invalid Username/Password. Invalid usernames and passwords consists of characters other than alphanumeric characters and the username already exists in the database.	System should not accept password and prevent user from creating a new user with a prompt to notify user.	Pass
TC3	Save user to device with enough memory	System should save the user information to device.	Pass

TC4	Save user to device with insufficient memory	Should not save user to device and prompt user not enough memory.	**Pass**
TC5	Tab cancel button	System should exit out of create user page and redirect back to login screen.	**Pass**

Test Cases: "Main Screen"

Project Name: **UTA Schedule Planner**
Test Case Name: **Main Screen**
Test Case Id: **CSE3310/Spring 2015/Team Psi/Main_Screen**

Test Case No.	Test Case Description	Expected results	Outcome Pass, Fail, Other (comments)
TC1	User has just logged in.	System should load all saved schedules on device.	**Pass**
TC2	Main screen is opened with no default schedule set.	Main screen should be displayed with no default schedule shown.	**Inconclusive**
TC3	Main screen is opened with a default schedule set.	Main screen should be displayed with the default schedule shown.	**Inconclusive**
TC4	Tab into User Settings.	System should direct to User Settings screen.	**Pass**
TC5	Tab Manage Schedule	System should direct to manage schedule screen.	**Pass**

TC6	Tab into Create Schedule	System should direct to create schedule screen.	**Pass**
TC7	Tab into Time Restrictions	System should direct user to time restrictions screen.	**Pass**
TC8	Tab into Log off	System should log off and return user to log in screen.	**Pass**

Test Cases: "Manage Schedules Screen"

Project Name: **UTA Schedule Planner;**
Test Case Name: **Main Schedules Screen**
Test Case Id: **CSE3310/Spring 2015/Team Psi/Main_Schedules_Screen**

Test Case No.	Test Case Description	Expected results	Outcome Pass, Fail, Other (comments)
TC1	Open manage schedules	Display list of schedules to choose from.	**Pass**
TC2	Tab into modify list of schedules with selected schedule	Should be directed to Create a Schedule with schedule settings.	**Inconclusive**
TC3	Tab into modify list of schedules with no schedules selected.	Should have nothing happen.	**Inconclusive**
TC4	Tab into "Set as main" button to set new main schedule	The selected schedule is now main schedule and should appear on main screen.	**Inconclusive**

TC5	Tab into "Cancel" button	System should exit out of manage schedules screen and redirect user back to main screen.	**Pass**
TC6	Tab into "Time Restrictions" button	System should exit out of manage schedules page and redirect user to time restriction screen.	**Removed**

Test Cases: "Time Restrictions Screen"

Project Name: **UTA Schedule Planner**
Test Case Name: **Time Restrictions Screen**
Test Case Id: **CSE3310/Spring 2015/Team Psi/Time_Restrictions_Screen**

Test Case No.	Test Case Description	Expected results	Outcome Pass, Fail, Other (comments)
TC1	Tab into start time/end time fields and times are valid. Done is selected. Valid hours is numbers 0-24 only. Valid minutes is **numbers 0-59 only.**	System should save time restrictions to device.	**Pass**
TC2	Tab into start time/end time fields and times are invalid. Done is selected. Invalid hours are characters other than	System should not save time restrictions and prompt notify user invalid time.	**Non-Applicable**

	numbers 0-24. Invalid minutes are characters other than numbers 0-59.		
TC3	Tab "Cancel" button	System should exit out of time schedules screen and redirect user back.	**Pass**

Test Cases: "User Settings Screen"

Project Name: **UTA Schedule Planner**
Test Case Name: **User Settings Screen**
Test Case Id: **CSE3310/Spring 2015/Team Psi/ User_Settings_Screen**

Test Case No.	Test Case Description	Expected results	Outcome Pass, Fail, Other (comments)
TC1	Tab into "Delete User" button with confirmation.	User should be deleted off the database and be direct to log in screen.	**Pass**
TC2	Tab into "Delete User" button then hitting cancel.	User should not be deleted off the database.	**Pass**
TC3	User enters in a new password with only alphanumeric characters in Change Password	Password should be changed for current User in the database.	**Pass**
TC4	User enters in a new password that are not alphanumeric in Change Password	Password should not be changed for current User in the database.	**Pass**

Test Cases: "Create Schedule Screen"

Project Name: UTA Schedule Planner
Test Case Name: User Settings Screen
Test Case Id: CSE3310/Spring 2015/Team Psi/ User_Settings_Screen

Test Case No.	Test Case Description	Expected results	Outcome Pass, Fail, Other (comments)
TC1	User chooses to load time restrictions.	System should redirect user to time restrictions page.	**Non-applicable**
TC2	User chooses to not load time restrictions.	System should continue on create schedule page.	**Non-applicable**
TC3	User has internet connection and enters valid course information in text fields and presses OK button. Valid information is alphanumeric characters only and course information that exists.	System should get course information for classes entered.	**Inconclusive**
TC4	User has internet connection and enters invalid course information in text fields and presses OK button. Valid information is alphanumeric characters. only and	System should not create a list of schedules and should prompt notify user.	**Inconclusive**

	course information that exists.		
TC5	User does not have internet connection and enters invalid course information in text fields and presses OK button. Invalid course information consists of characters other than alphanumeric and course information that do not exist.	System should not create a list of schedules and should prompt notify user.	**Inconclusive**
TC6	User does not have internet connection and enters valid course information in text fields and presses OK button. Valid information is alphanumeric characters only.	System should not create a list of schedules and should prompt notify user no internet connection available.	**Inconclusive**
TC5	System checks and there are no conflicts with loaded time restrictions	System should create a list of schedules.	**Inconclusive**
TC6	System checks and there are conflicts with loaded time restrictions	System should create a list of schedules excluding the conflicting courses..	**Inconclusive**
TC7	System checks for Saturday classes for cases TC5 and TC6	System should create a list including Saturday classes.	**Inconclusive**

TC8	User selects a schedule	System should save that schedule to the device.	**Inconclusive**
TC9	User selects back.	System should not save schedule to device. Redirect user to course information input.	**Pass**

6. Assumptions & Constraints

6.1 Assumptions

The following is a list of assumptions:

- Ignore network issues
- Ignore legal concerns
- Internet connection is provided
- All users will use Android 2.3 or higher
- All users are 18 or older and are UTA students

6.2 Constraints

The following is a list of constraints:

- Team has lack of Android development expertise
- Team has time conflicts.
- Team will be using Android Studio for development
- Team does not have available infrastructure for app

6.3 Out of Scope Material

The following is a list of "out of scope" material:

- Post project maintenance
- Risk assessment

7. Delivery & Schedule

Task/MileStone Description	Anticipated Start Date	Anticipated End Date	Status	Comments
Prepare Requirements and Use Cases	2/3/2015	2/26/2015	**Complete**	
SRA document (Includes project objective, Requirements and UML diagrams)	2/26/2015	3/31/2015	**Complete**	
Presentation of SRA	3/31/2015	3/31/2015	**Complete**	
Web page Design and Navigation	3/31/2015	4/23/2015	**Complete**	
Database Design	3/31/2015	4/23/2015	**Complete**	
Test Data Entry	3/31/2015	4/23/2015	**Complete**	
Test Plan Delivery	3/31/2015	4/23/2015	**Complete**	
... List all tasks ...				
External Documentation (i.e. User Manual)	4/23/2015	5/8/2015	**Complete**	
Final Milestone: Project delivery	4/23/2015	5/8/2015	**Complete**	

8. Stakeholder Approval Form

Stakeholder Name	Stakeholder Role	Stakeholder Comments	Stakeholder Approval Signature and Date
Thomas Hoang	**Project Assistant**		
Linh Vu	**Developer**		
Dennis Duong	**Developer**		
JR Ladera	**Development Mgr.**		

Appendix:

None.

UTA

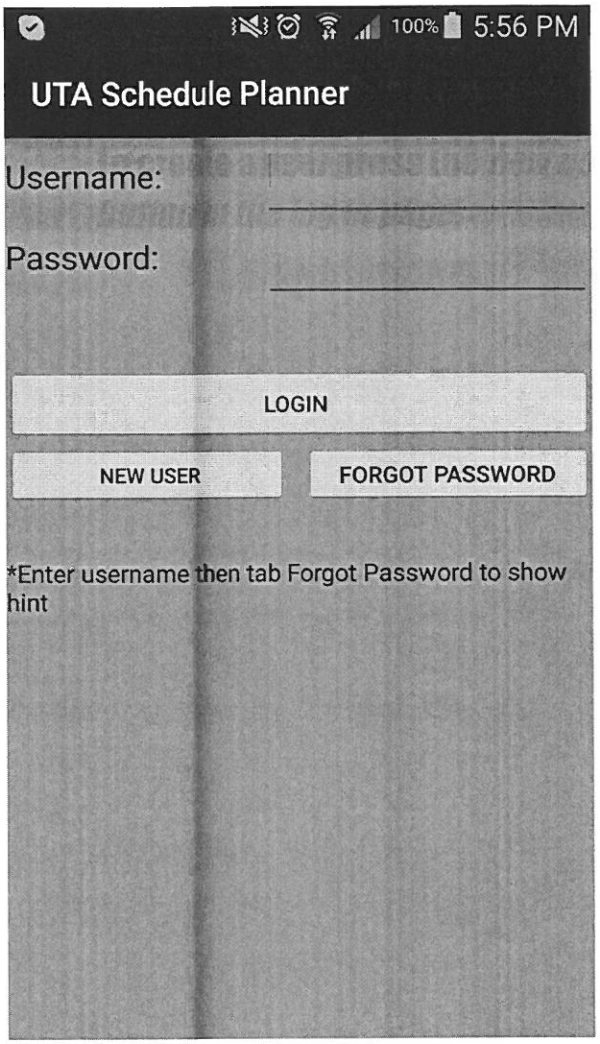
Schedule Planner

USER MANUAL

Table of Contents

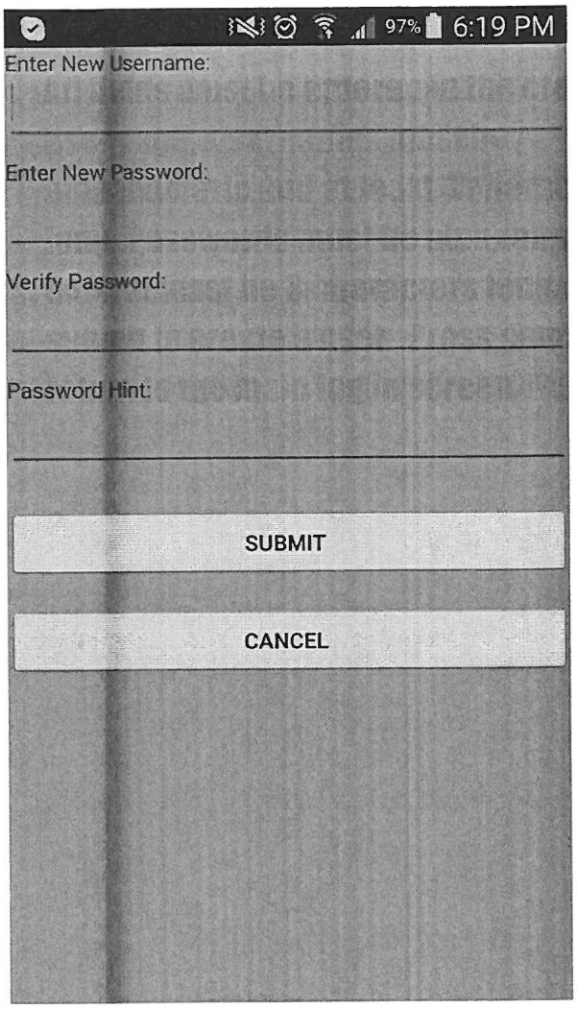
1. Logging in

After the application is loaded it will bring you to the main log in page. In order to log in a user must be created. To create a user press the *new user button in the log in page.*



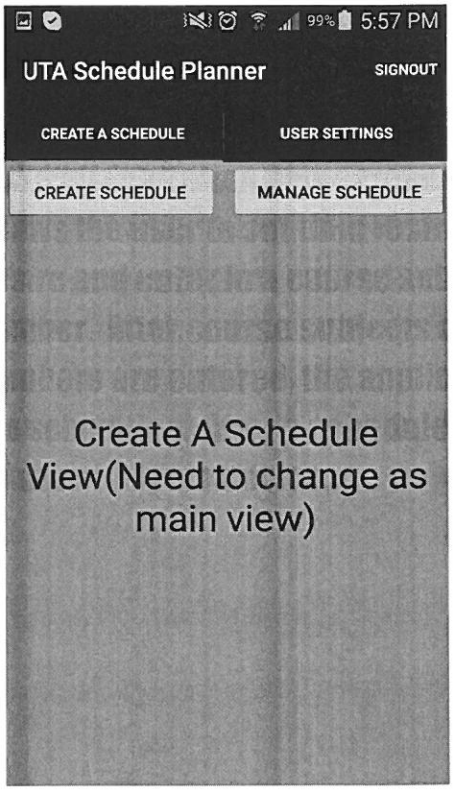
2. Create a User

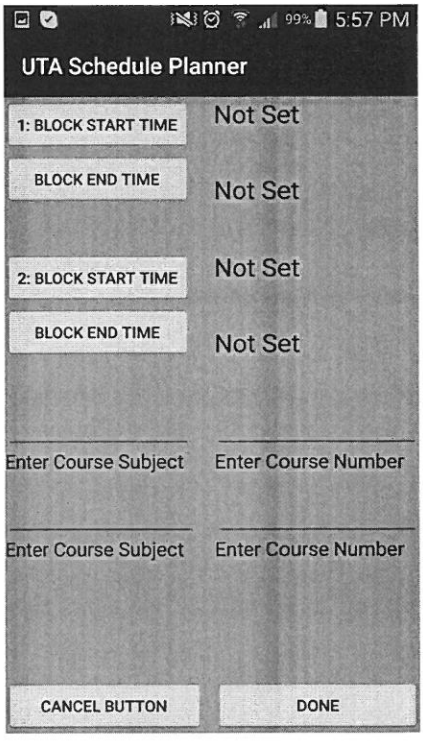
All fields must be entered in the create a user page. Usernames must be alphanumeric and at least 4 characters long. Passwords must be alphanumeric and must at least be 8 characters long. Press submit to create a user. Press cancel to return to the main log in screen.



3. Create a Schedule

To create a schedule press on the *CREATE SCHEDULE* button. A new screen will appear that allows the user to set time restrictions if you so desire and enter in a course subject and course number. After course subjects and course numbers are entered, the application will load the schedules. After a schedule is selected it will be saved to your device.





4. Manage Schedules

In manage schedules the user can set the selected schedule to be shown at the top of the screen.

The new schedule button allows the user to create another schedule.

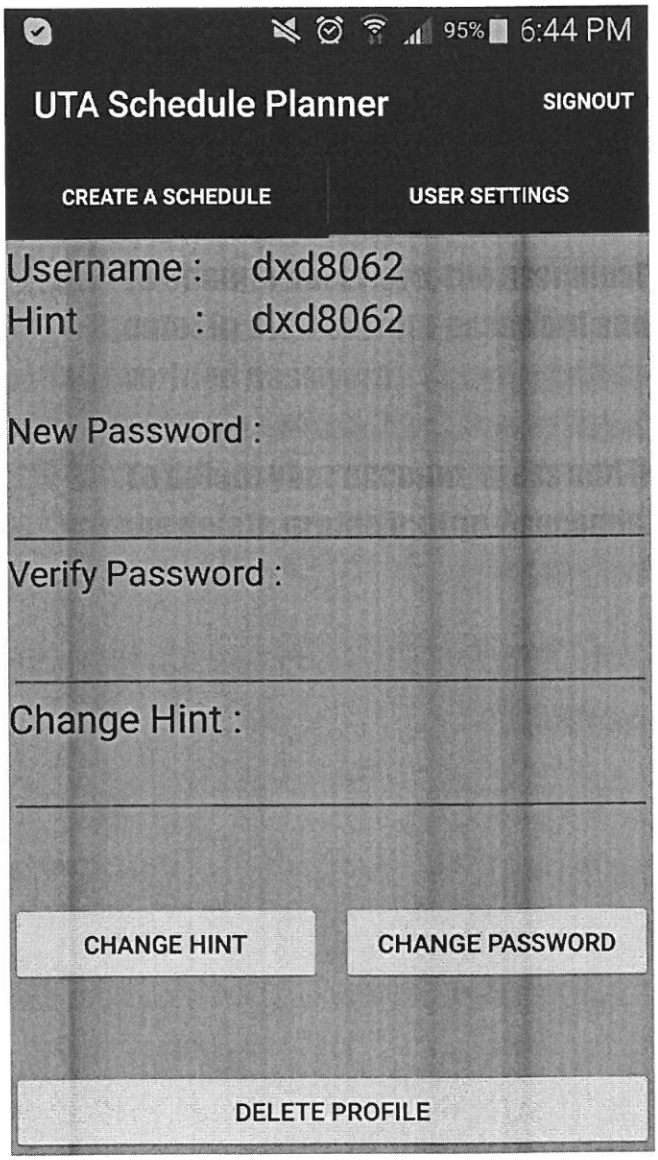
The *delete a schedule* allows the user to delete the selected schedule.

5. User Settings

The user settings page allows the user to change their password or delete their account.

To change password, the user must enter in their current password and verified password.

To delete your account, press and hold the delete profile button 2 seconds.



```
package com.psi.utascheduleplanner;

import android.content.Intent;
import android.os.Bundle;
import android.support.v7.app.ActionBarActivity;
import android.view.View;
import android.widget.Button;
import android.widget.Toast;

import java.sql.SQLException;

/**
 * Created by LagZ on 5/7/2015.
 */
public class ViewCreatedSchedule extends ActionBarActivity{

    UserDataBaseAdapter userDB;
    Button btnback2Create, btnSave;

    private String userName;

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_view_created_schedule);

        Intent intentInfo = getIntent();
        userName  = intentInfo.getStringExtra("username");

        userDB = new UserDataBaseAdapter(this);
        try {
            userDB.open();
        } catch (SQLException e) {
            e.printStackTrace();
        }

        btnback2Create = (Button) findViewById(R.id.btnBack);
        btnSave = (Button) findViewById(R.id.btnSave);

        //Take back to Create a schedule
        btnback2Create.setOnClickListener( new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                Intent nextScreen = new Intent(getApplicationContext(),
CreateSchedule.class);
                nextScreen.putExtra("username", userName);
                startActivity(nextScreen);
            }
        });

        //Once save take to manage schedule
        btnSave.setOnClickListener( new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                //Code here to save data to database
                Intent nextScreen = new Intent(getApplicationContext(),
ManageSchedule.class);
                nextScreen.putExtra("username", userName);
                startActivity(nextScreen);
            }
        });
    }
}
```

```
package com.psi.utascheduleplanner;

import android.content.Intent;
import android.os.Bundle;
import android.support.annotation.Nullable;
import android.support.v4.app.Fragment;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.Toast;

import java.sql.SQLException;
import java.util.regex.Matcher;
import java.util.regex.Pattern;

public class Usersettingsfragment extends Fragment implements View.OnClickListener,
View.OnLongClickListener {


    String userName = "TestName";
//Linh
    String viewHint ="TestHint";
    TextView seeName, seeHint;
    EditText newPassword, verifyPassword, newHintEnter;
    Button delButton,changePassBtn, changeHintBtn;

    UserDataBaseAdapter userDB;

    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup container,
                             Bundle savedInstanceState) {
        View view = inflater.inflate(R.layout.usersettingview, container, false);
        MainActivity mainInfo = (MainActivity) getActivity();

//LINH
        seeName = (TextView) view.findViewById(R.id.textUser);
        seeHint = (TextView) view.findViewById(R.id.showHint);

        newPassword = (EditText) view.findViewById(R.id.newPass);
        verifyPassword = (EditText) view.findViewById(R.id.verifyPass);
        newHintEnter = (EditText) view.findViewById(R.id.newHint);
//---------------------------------

        userDB = new UserDataBaseAdapter(view.getContext());

        try {
            userDB.open();
        } catch (SQLException e) {
            e.printStackTrace();
        }

        userName = mainInfo.getCurUser();
        viewHint = userDB.getHint(userName);
        seeHint.setText(viewHint);
        seeName.setText(userName);
        delButton = (Button) view.findViewById(R.id.delCurrentUser);
        changeHintBtn = (Button) view.findViewById(R.id.changeHint);
        changePassBtn = (Button) view.findViewById(R.id.changePass);
//LINH
        changePassBtn.setOnClickListener(this);
```

```
        changeHintBtn.setOnClickListener(this);

        delButton.setOnClickListener(this);
        delButton.setOnLongClickListener(this);
        return view;
    }

    @Override
    public void onClick(View view) {
        if(view.getId() == R.id.delCurrentUser)
        {
            Toast.makeText(getActivity(), "Press and hold to delete account. " +
userName, Toast.LENGTH_LONG).show();
        }
        else if(view.getId() == R.id.changeHint)
        {
            if(newHintEnter.length() == 0) {
                newHintEnter.setText("NO HINT MADE");
            }
            userDB.changeHint(userName, newHintEnter.getText().toString() );
            viewHint = userDB.getHint(userName);
            seeHint.setText(viewHint);
            Toast.makeText(getActivity(), "New hint updated",
Toast.LENGTH_SHORT).show();
        }
        else if(view.getId() == R.id.changePass)
        {
            final String checkPass = newPassword.getText().toString();

            if(!isValidEntry(checkPass)) {
                newPassword.requestFocus();
                newPassword.setError("Alphanumeric characters only");
            }
            else if(newPassword.length() == 0) {
                newPassword.requestFocus();
                newPassword.setError("Password field cannot be empty.");
            }

            else if(newPassword.length() < 8 || newPassword.length() > 15 ) {
                newPassword.requestFocus();
                newPassword.setError("Password can only be between 8 and 15
characters");
            }

            else
if(!verifyPassword.getText().toString().equals(newPassword.getText().toString())) {
                verifyPassword.requestFocus();
                verifyPassword.setError("Password does not match first input.");
            }
            else
            {
                userDB.changePassword(userName, newPassword.getText().toString());
                Toast.makeText(getActivity(), "Password successfully changed.",
Toast.LENGTH_SHORT).show();
            }
        }

    }

    @Override
    public boolean onLongClick(View view) {
        if(view.getId() == R.id.delCurrentUser){
            Toast.makeText(getActivity(), "Account deleted.",
```

```
Toast.LENGTH_LONG).show();

            userDB.delUser(userName);
            userDB.close();
            Intent loginScreen = new Intent(getActivity(), Login.class);
            startActivity(loginScreen);
            return true;
        }
        else
        {
            return false;
        }
    }

//LINH //Return true if characters in string is alphanumeric
    private boolean isValidEntry(String username) {
        String NAME_PATTERN = "[A-Za-z0-9]+";

        Pattern pattern = Pattern.compile(NAME_PATTERN);
        Matcher matcher = pattern.matcher(username);
        return matcher.matches();
    }

    @Override
    public void onDestroy() {
        super.onDestroy();
        userDB.close();
    }


}
```

```
package com.psi.utascheduleplanner;

import android.content.ContentValues;
import android.content.Context;
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;

import java.sql.SQLException;

/**
 * Created by laosd_000 on 5/1/2015.
 */

public class UserDataBaseAdapter {
    static final String DATABASE_NAME = "users.db";

    static final int DATABASE_VERSION = 1;

    public static final int USER_COLUMN = 1;
    // TODO: Create public field for each column in your table.
    // SQL Statement to create a new database.
//LINH on HINT text
    static final String DATABASE_CREATE = "create table "+"LOGIN"+
            "( " +"ID"+" integer primary key autoincrement,"+ "USERNAME
text,PASSWORD text,HINT text, " +
            "STARTBLOCK1 text, ENDBLOCK1 text, STARTBLOCK2 text, ENDBLOCK2 text); ";

    public SQLiteDatabase sqDB;
    private final Context context;
    private DatabaseHelper dbHelper;

    public UserDataBaseAdapter(Context thatContext) {
        context = thatContext;
        dbHelper = new DatabaseHelper(context,DATABASE_NAME,null,DATABASE_VERSION);
    }

    //Method opens database
    public UserDataBaseAdapter open() throws SQLException {
        sqDB = dbHelper.getWritableDatabase();
        return this;
    }

    //Method will close database
    public void close() {
        sqDB.close();
    }

    //Method will return instance of the database
    public SQLiteDatabase getDatabaseInstance() {
        return sqDB;
    }

    //Method will insert the new user info into the database for creation
//LINH on Sring hint parameter
    public void insertEntry(String username, String password, String hint) {
        ContentValues newValues = new ContentValues();
        newValues.put("USERNAME", username);
        newValues.put("PASSWORD", password);
//LINH
        newValues.put("HINT", hint);

        //Inserting the row into table here
        sqDB.insert("LOGIN",null,newValues);
```

```
    }

    //LINH need to check this
    public void insertTime(String startBlock1, String endBlock1, String startBlock2,
String endBlock2) {
        ContentValues newValues = new ContentValues();
        newValues.put("STARTBLOCK1", startBlock1);
        newValues.put("ENDBLOCK1", endBlock1);
        newValues.put("STARTBLOCK2", startBlock2);
        newValues.put("ENDBLOCK2", endBlock2);

        //Inserting the row into table here

//        String where="USERNAME = ?";
//        sqDB.update("LOGIN", newValues, where, new String[]{username});

        sqDB.insert("LOGIN",null,newValues);
    }

    //Method to delete a user
    public boolean delUser(String username) {
        String where = "USERNAME=?";
        //int entryNum= sqDB.delete("LOGIN", where, new String[]{username}) ;
        return sqDB.delete("LOGIN", where, new String[]{username}) > 0;
    }

    //Method to return a password of a user
    public String getPassword(String username) {
        Cursor cursor = sqDB.query("LOGIN", null, "USERNAME=?", new
String[]{username}, null, null, null);
        //If username does not exist
        if(cursor.getCount()<1)
            return "NOT EXIST";
        cursor.moveToFirst();
        String password= cursor.getString(cursor.getColumnIndex("PASSWORD"));
        return password;
    }

//Linh //Method return password hint
    public String getHint(String username) {
        Cursor cursor = sqDB.query("LOGIN", null, "USERNAME=?", new
String[]{username}, null, null, null);
        //If username does not exist
        if(cursor.getCount()<1)
            return "NO HINT CREATED";
        cursor.moveToFirst();
        String hint = cursor.getString(cursor.getColumnIndex("HINT"));
        return hint;
    }

//LINH //Method return username
    public String getUser(String username) {
        Cursor cursor = sqDB.query("LOGIN", null, "USERNAME=?", new
String[]{username}, null, null, null);
        //If username does not exist
        if(cursor.getCount()<1)
            return "NOT EXIST";
        cursor.moveToFirst();
        String name = cursor.getString(cursor.getColumnIndex("USERNAME"));
        return name;
    }

    //Method to update user info, only change password for now
```

```
    public void changePassword(String username, String password) {
        ContentValues updateValues = new ContentValues();

        updateValues.put("PASSWORD", password);

        String where="USERNAME = ?";
        sqDB.update("LOGIN", updateValues, where, new String[]{username});
    }

//LINH
    //Method to update user info, only change password for now
    public void changeHint(String username, String hint) {
        ContentValues updateValues = new ContentValues();

        updateValues.put("HINT", hint);

        String where="USERNAME = ?";
        sqDB.update("LOGIN", updateValues, where, new String[]{username});
    }
}
```

```
package com.psi.utascheduleplanner;

/**
 * Created by Dennis on 4/27/2015.
 */

import android.support.v4.app.Fragment;
import android.support.v4.app.FragmentManager;
import android.support.v4.app.FragmentStatePagerAdapter;

public class Tabsadapter  extends FragmentStatePagerAdapter{

    private int TOTAL_TABS = 2;

    public Tabsadapter(FragmentManager fm) {
        super(fm);
        // TODO Auto-generated constructor stub
    }

    @Override
    public Fragment getItem(int index) {
        // TODO Auto-generated method stub
        switch (index) {
            case 0:
                return new Createfragment();

            case 1:
                return new Usersettingsfragment();

        }

        return null;
    }

    @Override
    public int getCount() {
        // TODO Auto-generated method stub
        return TOTAL_TABS;
    }

}
```

```
package com.psi.utascheduleplanner;

import android.content.Intent;
import android.os.Bundle;
import android.support.v7.app.ActionBarActivity;
import android.view.View;
import android.widget.Button;
import android.widget.Toast;

import java.sql.SQLException;

/**
 * Created by LagZ on 5/6/2015.
 */
public class ManageSchedule extends ActionBarActivity{

    UserDataBaseAdapter userDB;
    Button deleteSched, newSchedule, setMainSched, btnBack;

    private String userName;
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_manage_schedule);

        Intent intentInfo = getIntent();
        userName  = intentInfo.getStringExtra("username");

        deleteSched = (Button) findViewById(R.id.delSched);
        newSchedule = (Button) findViewById(R.id.newSched);
        setMainSched = (Button) findViewById(R.id.mainSched);
        btnBack = (Button) findViewById(R.id.back);



        //Deleting a schedule
        deleteSched.setOnClickListener( new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                Toast.makeText(ManageSchedule.this, "Deleting Schedule",
Toast.LENGTH_SHORT).show();
            }
        });

        //Set chosen schedule as main
        setMainSched.setOnClickListener( new View.OnClickListener() {
            @Override
            public void onClick(View view) {
               //Store new main schedule here
                Toast.makeText(ManageSchedule.this, "Setting new main schedule",
Toast.LENGTH_SHORT).show();
            }
        });

        //Button to create new schedule
        newSchedule.setOnClickListener( new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                Intent nextScreen = new Intent(getApplicationContext(),
CreateSchedule.class);
                nextScreen.putExtra("username", userName);
                startActivity(nextScreen);
            }
```

```
package com.psi.utascheduleplanner;

import android.content.Intent;
import android.support.v7.app.ActionBarActivity;
import android.support.v7.app.ActionBar;
import android.support.v7.app.ActionBar.Tab;
import android.os.Bundle;
import android.view.Menu;
import android.view.MenuItem;
import android.support.v4.app.FragmentTransaction;
import android.support.v4.view.ViewPager;
import android.support.v4.view.ViewPager.OnPageChangeListener;
import android.widget.Toast;

//MainActivity consists of the tabs which will create create a sched, manage
schedule, etc.
public class MainActivity extends ActionBarActivity implements
android.support.v7.app.ActionBar.TabListener {

    private ViewPager tabsviewPager;
    private ActionBar mActionBar;
    private Tabsadapter mTabsAdapter;
    private String curUser;
    private String curPassword;
    private String curHint;

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        getMenuInflater().inflate(R.menu.menu_main, menu);
        return true;
    }

    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        int id = item.getItemId();
        if (id == R.id.action_signout) {
            Intent nextScreen = new Intent(getApplicationContext(), Login.class);
            startActivity(nextScreen);
            finish();
            return true;
        }
        return super.onOptionsItemSelected(item);
    }

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        Intent intentInfo = getIntent();

        String userName = intentInfo.getStringExtra("username");
        String userPassword = intentInfo.getStringExtra("password");
        String userHint = intentInfo.getStringExtra("hint");


        curUser = userName;
        curPassword = userPassword;
        curHint = userHint;

        tabsviewPager = (ViewPager) findViewById(R.id.tabspager);

        mTabsAdapter = new Tabsadapter(getSupportFragmentManager());
```

```
        tabsviewPager.setAdapter(mTabsAdapter);

        getSupportActionBar().setHomeButtonEnabled(false);
        getSupportActionBar().setNavigationMode(ActionBar.NAVIGATION_MODE_TABS);

        Tab createtab = getSupportActionBar().newTab().setText("Create A
Schedule").setTabListener(this);
        Tab usersettingtab = getSupportActionBar().newTab().setText("User
Settings").setTabListener(this);

        getSupportActionBar().addTab(createtab);
        getSupportActionBar().addTab(usersettingtab);

        //This helps in providing swiping effect for v7 compat library
        tabsviewPager.setOnPageChangeListener(new OnPageChangeListener() {

            @Override
            public void onPageSelected(int position) {
                // TODO Auto-generated method stub
                getSupportActionBar().setSelectedNavigationItem(position);
            }

            @Override
            public void onPageScrolled(int arg0, float arg1, int arg2) {
                // TODO Auto-generated method stub

            }

            @Override
            public void onPageScrollStateChanged(int arg0) {
                // TODO Auto-generated method stub

            }
        });

    }


    public String getCurUser() {
        return curUser;
    }

    public String getCurPassword() {
        return curPassword;
    }

    public String getCurHint() { return curHint; }

    @Override
    public void onBackPressed() {
        //Do nothing
    }

    @Override
    public void onTabReselected(Tab arg0, FragmentTransaction arg1) {
        // TODO Auto-generated method stub

    }

    @Override
    public void onTabSelected(Tab selectedtab, FragmentTransaction arg1) {
        // TODO Auto-generated method stub
```

```
        tabsviewPager.setCurrentItem(selectedtab.getPosition()); //update tab
position on tap
    }

    @Override
    public void onTabUnselected(Tab arg0, FragmentTransaction arg1) {
        // TODO Auto-generated method stub

    }
}
```

```
package com.psi.utascheduleplanner;

import android.app.ActionBar;
import android.app.Activity;
import android.content.Intent;
import android.os.Bundle;
import android.support.v7.app.ActionBarActivity;
import android.util.Log;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;

import java.sql.SQLException;

public class Login extends ActionBarActivity {
    UserDataBaseAdapter userDB;

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_login);
//DELETE ALL DATABASE use for testing only
//        this.deleteDatabase("users.db");

        userDB = new UserDataBaseAdapter(this);
        try {
            userDB = userDB.open();
        } catch (SQLException e) {
            e.printStackTrace();
        }

        final EditText inputName = (EditText) findViewById(R.id.name);
        final EditText inputPassword = (EditText) findViewById(R.id.password);

//LINH
        Button btnForgotPass = (Button) findViewById(R.id.btnForgotPassword);
//        Button test = (Button) findViewById((R.id.testButton));

        Button btnNextScreen = (Button) findViewById(R.id.btnNextScreen);
        Button newUserButton = (Button) findViewById(R.id.newUserButton);

        //If login successful, go to main page
        btnNextScreen.setOnClickListener(new View.OnClickListener() {

            public void onClick(View arg0) {
                //Take in entered values
                String enteredName = inputName.getText().toString();
                String enteredPassword = inputPassword.getText().toString();

                //Fetch stored password from database of given username
                String storedPassword = userDB.getPassword(enteredName);

                if(storedPassword.equals(enteredPassword)) {
                    Intent nextScreen = new Intent(getApplicationContext(),
MainActivity.class);
                    Toast.makeText(getApplicationContext(), "Log in successful.",
Toast.LENGTH_SHORT).show();
                    //Sending username and password into next intent for use
                    nextScreen.putExtra("username", enteredName);
                    nextScreen.putExtra("password", storedPassword);
                    Toast.makeText(getApplicationContext(), "Welcome " + enteredName
+ ".", Toast.LENGTH_SHORT).show();
```

```
                    inputName.getText().clear();
                    inputPassword.getText().clear();
                    startActivity(nextScreen);
                }
                else {
                    Toast.makeText(getApplicationContext(), "Username and password
does not match.", Toast.LENGTH_LONG).show();
                    inputName.getText().clear();
                    inputPassword.getText().clear();
                }
            }
        });

        //Create a new user
        newUserButton.setOnClickListener(new View.OnClickListener() {

            public void onClick(View arg0) {
                Intent nextScreen = new Intent(getApplicationContext(),
CreateUser.class);
                startActivity(nextScreen);
                finish();
            }
        });

//LINH
        btnForgotPass.setOnClickListener(new View.OnClickListener() {

            public void onClick(View arg0) {
                String enteredName = inputName.getText().toString();
                Toast.makeText(Login.this, userDB.getHint(enteredName),
Toast.LENGTH_SHORT).show();
            }
        });

////LINH for testing activity!!
//        test.setOnClickListener(new View.OnClickListener() {
//
//            public void onClick(View arg0) {
//                Intent nextScreen = new Intent(getApplicationContext(),
CreateSchedule.class);
//                startActivity(nextScreen);
//                finish();
//            }
//        });
    }
}
```

```
package com.psi.utascheduleplanner;

import android.content.Context;
import android.database.sqlite.SQLiteDatabase;
import android.database.sqlite.SQLiteDatabase.CursorFactory;
import android.database.sqlite.SQLiteOpenHelper;
import android.util.Log;

public class DatabaseHelper extends SQLiteOpenHelper {
    public DatabaseHelper(Context context, String name,CursorFactory factory, int
version) {
        super(context, name, factory, version);
    }

    //This will be called when no database exists
    @Override
    public void onCreate(SQLiteDatabase db) {
        db.execSQL(UserDataBaseAdapter.DATABASE_CREATE);
    }

    //This is called when version types of the databases do not match and need to be
upgraded
    @Override
    public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {
        //Create a log about the version upgrade
        //Will destroy old data
        Log.w("DBAdapter", " upgrading from " + oldVersion + " to " + newVersion +
".");
        db.execSQL("Drop table if it exists");
        onCreate(db);
    }
}
```

```
package com.psi.utascheduleplanner;

import android.app.Activity;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;

import java.sql.SQLException;
import java.util.regex.Matcher;
import java.util.regex.Pattern;

public class CreateUser extends Activity {
    // Initializing variables
    EditText inputName;
    EditText inputPassword;
    EditText inputVerifyPassword;
//LINH
    EditText inputHint;

    UserDataBaseAdapter newUser;

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_create_user);

        newUser = new UserDataBaseAdapter(this);
        try {
            newUser = newUser.open();
        } catch (SQLException e) {
            e.printStackTrace();
        }

        inputName = (EditText) findViewById(R.id.name);
        inputPassword = (EditText) findViewById(R.id.password);
        inputVerifyPassword = (EditText) findViewById(R.id.passwordV);
//LINH
        inputHint = (EditText) findViewById(R.id.textHint);

        Button btnNextScreen = (Button) findViewById(R.id.btnNextScreen);
        Button cancelButton = (Button) findViewById(R.id.btnCancel);

        //Listening to button event
        btnNextScreen.setOnClickListener(new View.OnClickListener() {

            public void onClick(View arg0) {
                //Validating username and password
                //LINH
                if(inputHint.length() == 0) {
                    inputHint.setText("NO HINT MADE");
                }

                final String checkName =
newUser.getUser(inputName.getText().toString());
                final String newName = inputName.getText().toString();
                final String checkPass = inputPassword.getText().toString();
//----------------------------------------


                //Validating username and password
```

```
                if(inputName.length() == 0) {
                    inputName.requestFocus();
                    inputName.setError("Username field cannot be empty.");
                }
//LINH
                else if(inputName.length() < 4 || inputName.length() > 12) {
                    inputName.requestFocus();
                    inputName.setError("Username can only be between 4 and 12
characters");
                }

                else if(newName.equals(checkName))
                {
                    inputName.requestFocus();
                    inputName.setError("Username already exist.");
                }
                else if(!isValidEntry(newName)) {
                    inputName.requestFocus();
                    inputName.setError("Alphanumeric characters only");
                }
                else if(!isValidEntry(checkPass)) {
                    inputPassword.requestFocus();
                    inputPassword.setError("Alphanumeric characters only");
                }
//------------------------------

                else if(inputPassword.length() == 0) {
                    inputPassword.requestFocus();
                    inputPassword.setError("Password field cannot be empty.");
                }
//LINH
                else if(inputPassword.length() < 8 || inputPassword.length() > 15 )
{
                    inputPassword.requestFocus();
                    inputPassword.setError("Password can only be between 8 and 15
characters");
                }

                else
if(!inputVerifyPassword.getText().toString().equals(inputPassword.getText().toString
())) {
                    inputVerifyPassword.requestFocus();
                    inputVerifyPassword.setError("Password does not match first
input.");
                }
//LINH on inputHint.getText().toString() last parameter
                else {
                    newUser.insertEntry(inputName.getText().toString(),
                                        inputPassword.getText().toString(),
                                            inputHint.getText().toString());
                    Toast.makeText(getApplicationContext(), "User created.",
Toast.LENGTH_SHORT).show();
                    Intent nextScreen = new Intent(getApplicationContext(),
Login.class);
                    startActivity(nextScreen);
                }

            }
        });

        cancelButton.setOnClickListener(new View.OnClickListener() {

            public void onClick(View arg0) {
```

```
                //Cancel will return to main log in screen
                Intent nextScreen = new Intent(getApplicationContext(),
Login.class);
                startActivity(nextScreen);
                }


        });
    }

//LINH //Return true if characters in string is alphanumeric
    private boolean isValidEntry(String username) {
        String NAME_PATTERN = "[A-Za-z0-9]+";

        Pattern pattern = Pattern.compile(NAME_PATTERN);
        Matcher matcher = pattern.matcher(username);
        return matcher.matches();
    }
//------------------------------

    @Override
    public void onBackPressed() {
        //Do nothing, disabling back button on android to prevent from returning to
pages incorrectly
    }

    @Override
    protected void onDestroy() {
        super.onDestroy();
        newUser.close();
    }
}
```

```
package com.psi.utascheduleplanner;

import android.app.Dialog;
import android.app.TimePickerDialog;
import android.content.Intent;
import android.os.Bundle;
import android.support.v7.app.ActionBarActivity;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.TimePicker;
import android.widget.Toast;

import java.sql.SQLException;

/**
 * Created by LagZ on 5/6/2015.
 */
public class CreateSchedule extends ActionBarActivity {

    UserDataBaseAdapter userDB;

    Button btnStartBlock1, btnEndBlock1;
    Button btnStartBlock2, btnEndBlock2;
    Button cancelButton, finishButton;
    int hour_x;
    int minute_x;
    String am_pm, change;
    String start1, end1, start2, end2;

    private String userName;

    TextView viewStartTime1, viewStartTime2, viewEndTime1, viewEndTime2;
    EditText inputSubject, inputCourseNum;

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_create_schedule);

        userDB = new UserDataBaseAdapter(this);
        try {
            userDB.open();
        } catch (SQLException e) {
            e.printStackTrace();
        }

        Intent intentInfo = getIntent();
        userName  = intentInfo.getStringExtra("username");

        viewStartTime1 = (TextView) findViewById(R.id.textView1);
        viewEndTime1 = (TextView) findViewById(R.id.textView2);
        viewStartTime2 = (TextView) findViewById(R.id.textView3);
        viewEndTime2 = (TextView) findViewById(R.id.textView4);

        inputSubject = (EditText) findViewById(R.id.editCourseSubj1);
        inputCourseNum = (EditText) findViewById(R.id.editCourseNum1);

        showTimePickerDialog();
    }

    //Pushing the start and end button to show up time dialog to set up block time
```

```
    public void showTimePickerDialog() {
        btnStartBlock1 = (Button) findViewById(R.id.startBlock1);
        btnEndBlock1 = (Button) findViewById(R.id.endBlock1);
        btnStartBlock2 = (Button) findViewById(R.id.startBlock2);
        btnEndBlock2 = (Button) findViewById(R.id.endBlock2);
        cancelButton = (Button) findViewById(R.id.btnCancel);
        finishButton = (Button) findViewById(R.id.btnFinish);



        btnStartBlock1.setOnClickListener(
                new View.OnClickListener() {
                    @Override
                    public void onClick(View view) {
                        showDialog(1);
                    }
                }
        );

        btnEndBlock1.setOnClickListener(
                new View.OnClickListener() {
                    @Override
                    public void onClick(View view) {
                        showDialog(2);
                    }
                }
        );

        btnStartBlock2.setOnClickListener(
                new View.OnClickListener() {
                    @Override
                    public void onClick(View view) {
                        showDialog(3);
                    }
                }
        );
        btnEndBlock2.setOnClickListener(
                new View.OnClickListener() {
                    @Override
                    public void onClick(View view) {
                        showDialog(4);

                    }
                }
        );
        //Cancel
        cancelButton.setOnClickListener(new View.OnClickListener() {

            public void onClick(View arg0) {
                Intent nextScreen = new Intent(getApplicationContext(),
ManageSchedule.class);
                nextScreen.putExtra("username", userName);
                startActivity(nextScreen);
            }
        });

        //On finish this will go to a function to create the schedule
        finishButton.setOnClickListener(new View.OnClickListener() {

            public void onClick(View arg0) {
                Intent nextScreen = new Intent(getApplicationContext(),
ViewCreatedSchedule.class);
                nextScreen.putExtra("username", userName);
```

```
                startActivity(nextScreen);
                userDB.insertTime(start1, end1, start2, end2);
                Toast.makeText(CreateSchedule.this, "Creating",
Toast.LENGTH_LONG).show();
            }
        });

    }

    @Override
    public Dialog onCreateDialog(int id) {
        if(id == 1)
        {
            return new TimePickerDialog(CreateSchedule.this, startTimeListener1,
hour_x, minute_x, false);
        }
        else if(id == 2)
        {
            return new TimePickerDialog(CreateSchedule.this, endTimeListener1,
hour_x, minute_x, false);
        }
        else if(id == 3)
        {
            return new TimePickerDialog(CreateSchedule.this, startTimeListener2,
hour_x, minute_x, false);
        }
        else if(id == 4)
        {
            return new TimePickerDialog(CreateSchedule.this, endTimeListener2,
hour_x, minute_x, false);
        }
        else
        {
            return null;
        }


    }

    protected TimePickerDialog.OnTimeSetListener startTimeListener1 =
            new TimePickerDialog.OnTimeSetListener() {
                @Override
                public void onTimeSet(TimePicker view, int hourOfDay, int minute) {
                    repeat(hourOfDay, minute);
                    start1 = change;
                    viewStartTime1.setText(change);
                }
            };

    protected TimePickerDialog.OnTimeSetListener endTimeListener1 =
            new TimePickerDialog.OnTimeSetListener() {
                @Override
                public void onTimeSet(TimePicker view, int hourOfDay, int minute) {
                    repeat(hourOfDay, minute);
                    end1 = change;
                    viewEndTime1.setText(change);
                }
            };

    protected TimePickerDialog.OnTimeSetListener startTimeListener2 =
            new TimePickerDialog.OnTimeSetListener() {
                @Override
                public void onTimeSet(TimePicker view, int hourOfDay, int minute) {
```

```
package com.psi.utascheduleplanner;

/**
 * Created by laosd_000 on 4/28/2015.
 */

import android.content.Intent;
import android.os.Bundle;
import android.support.v4.app.Fragment;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.Button;

import java.sql.SQLException;


public class Createfragment extends Fragment {

    Button buttonCreateSched, buttonManageSched;
//LINH
   String userName;

    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup container,
                             Bundle savedInstanceState) {
        View view = inflater.inflate(R.layout.createview, container, false);
//LINH EVERYTHING
        MainActivity mainInfo = (MainActivity) getActivity();
        buttonCreateSched = (Button) view.findViewById(R.id.btnCreateSched);
        buttonManageSched = (Button) view.findViewById(R.id.btnManageSched);

        userName = mainInfo.getCurUser();

        //Button to create
        buttonCreateSched.setOnClickListener( new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                Intent nextScreen = new Intent(getActivity(), CreateSchedule.class);
                nextScreen.putExtra("username", userName);
                startActivity(nextScreen);
            }
        });

        //Button to manage
        buttonManageSched.setOnClickListener( new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                Intent nextScreen = new Intent(getActivity(), ManageSchedule.class);
                nextScreen.putExtra("username", userName);
                startActivity(nextScreen);
            }
        });




        return view;
    }
}
```

```
        });

        //Back button to main screen
        btnBack.setOnClickListener( new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                Intent nextScreen = new Intent(getApplicationContext(),
MainActivity.class);
                nextScreen.putExtra("username", userName);
                startActivity(nextScreen);
            }
        });
    }


}
```