TRƯỜNG CAO ĐẲNG CÔNG NGHỆ THỦ ĐỨC
KHOA CÔNG NGHỆ THÔNG TIN

BÁO CÁO KẾT THÚC MÔN HỌC

Lập trình Android nâng cao

XÂY DỰNG ỨNG DỤNG "WORK MANAGER"

Giảng viên hướng dẫn: Tiêu Kim Cương

Sinh viên thực hiện:

1. Nguyễn Phương Hiếu
2. Nguyễn Anh Hoan
3. Bùi Nguyễn Sơn Tùng
4. Nguyễn Thị Mai Ly
5. Võ Minh Châu
6. Nguyễn Ngọc Xuân Anh

Ngành: Công nghệ thông tin Khoá: 2014

Tp. Hồ Chí Minh, Ngày 21 Tháng 12 Năm 2016

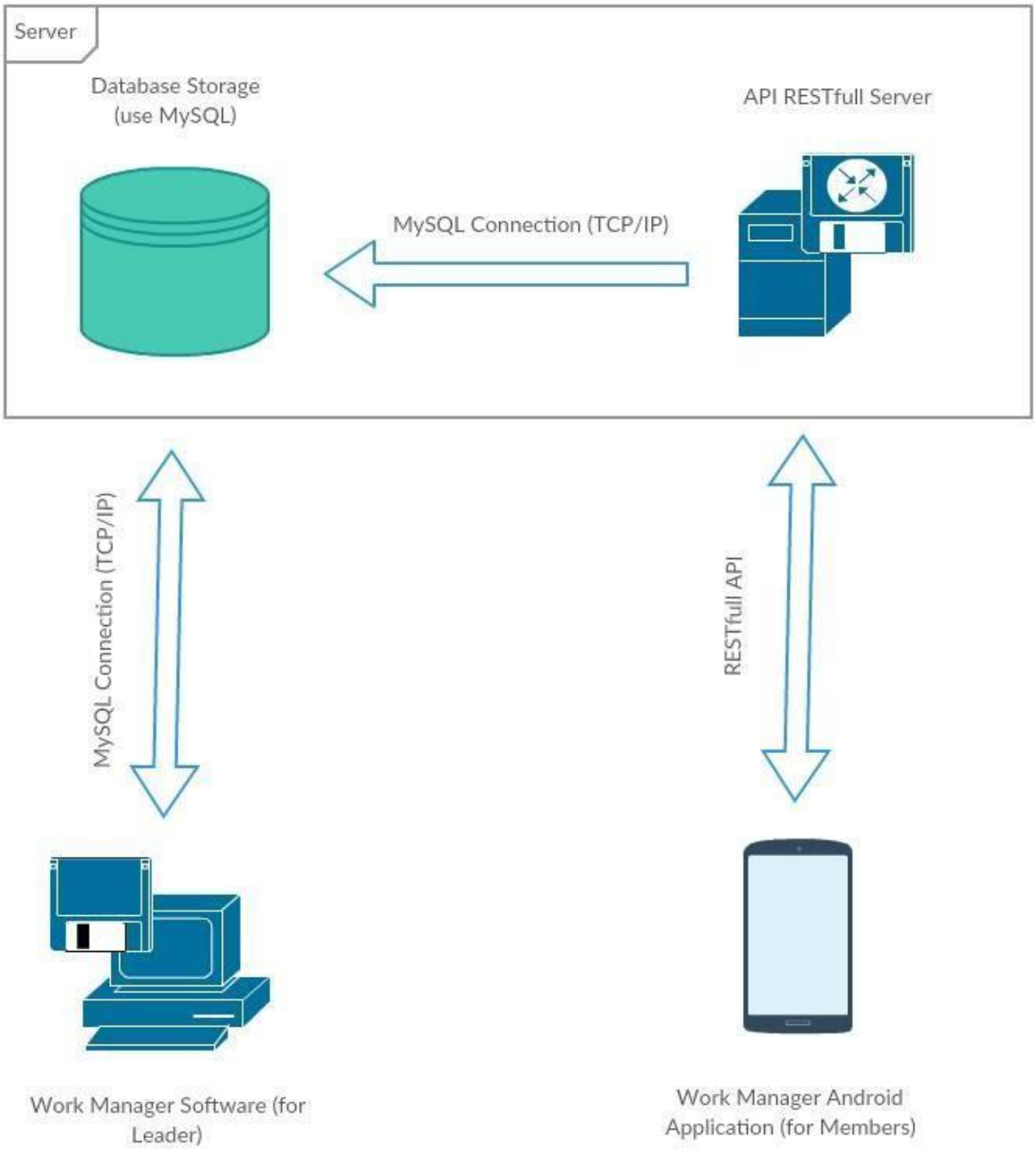
NHẬT KÝ HOẠT ĐỘNG NHÓM

Stt	Họ và tên	Công việc đã thực hiện	Tự đánh giá	Nhóm đánh giá	Chữ ký
1	Nguyễn Phương Hiếu	Nghiên cứu đề tài Thiết kế cơ sở dữ liệu Viết SDS Viết SRS Thiết kế UI/UX server manager Thiết kế công cụ Anti-Empty directory Server Manager App Test SRS Design WebAPI Hỗ trợ cho nhóm và tổng hợp App			
2	Nguyễn Anh Hoan	Nghiên cứu đề tài Viết SDS Viết SRS Cập nhật SDS, SRS Parse JSON Xử lý tại Task List Fragment Linking Core Class Hỗ trợ cho nhóm			
3	Bùi Nguyễn Sơn Tùng	Nghiên cứu đề tài Thiết kế UI trên Pencil Viết SRS Thiết kế UI/UX mobile Get Phone Number AsyncTask Loading Activity Xử lý tại Detail Fragment Hỗ trợ cho nhóm			

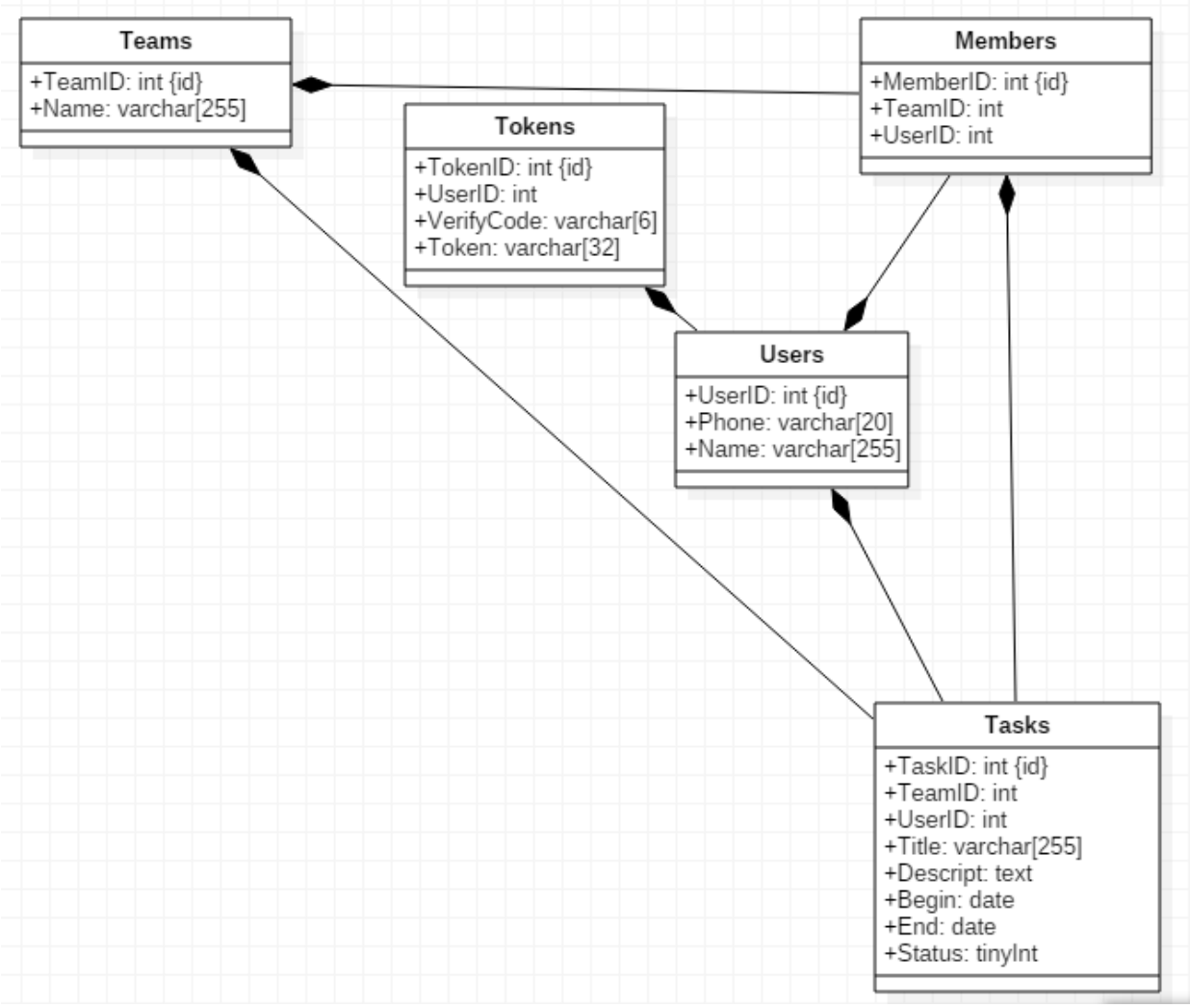
4	Nguyễn Thị Mai Ly	Nghiên cứu đề tài Viết SRS Viết Test Cases Test SRS Thiết kế UI/UX mobile List View Custom Item Adapter Test Mobile App ver.1 Báo cáo cuối kỳ Xử lý tại Task List Activity Update Manifest Run testing			
5	Võ Minh Châu	Nghiên cứu đề tài Viết SRS Thiết kế UI/UX mobile Thiết kế Fragment Adapter Báo cáo cuối kỳ Xử lý tại Task List Activity Thiết kế Custom List View Item			
6	Nguyễn Ngọc Xuân Anh	Nghiên cứu đề tài Viết SRS Viết Test cases Tạo Core Class Test Mobile App ver.1 Xử lý tại Loading Activity Báo cáo cuối kỳ			

MỤC LỤC

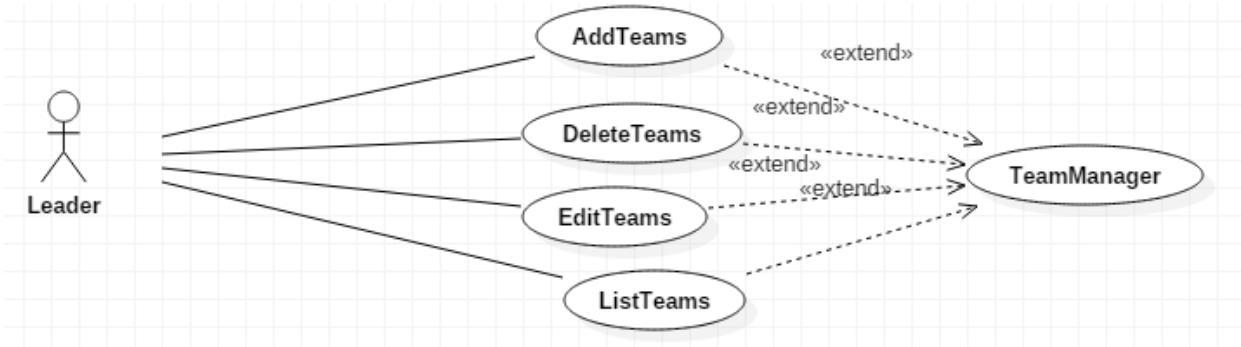
DANH MỤC BẢNG BIỂU, HÌNH VẼ, SƠ ĐỒ



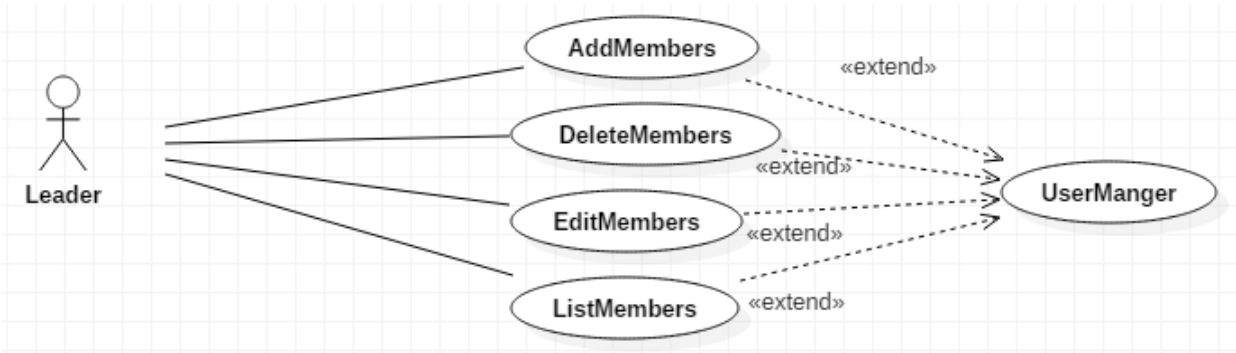
1.1 Model Overview



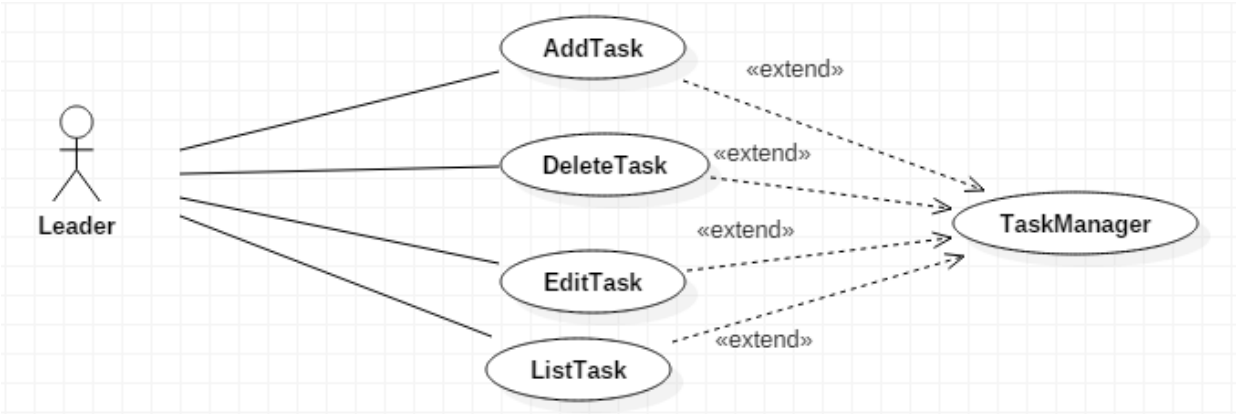
1.2. Database diagrams



1.3 Use case diagram - Teams



1.4 Use case diagram – Member



1.5 Use case diagram - Task

CHƯƠNG 1. MỞ ĐẦU

1. Giới thiệu môn học và nhóm thực hiện

Sau khi kết thúc Lập trình di động 1 sinh viên sẽ tiếp tục với Lập trình Di động 2. Với môn học này sinh viên tiếp tục phát triển khả năng lập trình, nâng cao khả năng tạo ra ứng dụng sử dụng trong thực tế.

Sinh viên sẽ được học cách tạo ra các Fragment cho một layout, kỹ thuật sử dụng Service để chạy ngầm ứng dụng, cũng như sử dụng AsyncTask có khả năng chạy nhiều công việc một lúc…

Với kiến thức cũng như được học thêm từ môn học Lập trình Di động 2, chúng tôi đã cùng nhau tạo ra một ứng dụng Work Manager để quản lý công việc. Thành viên tham gia thực hiện đề tài:

- Nguyễn Phương Hiếu (Trưởng nhóm).
- Nguyễn Anh Hoan.
- Bùi Nguyễn Sơn Tùng.
- Nguyễn Ngọc Xuân Anh.
- Nguyễn Thị Mai Ly.
- Võ Minh Châu.

2. Mô tả ứng dụng

Hệ thống ứng dụng gồm có 3 phần:

- Ứng dụng trên android. (Người dùng)
- Phần mềm trên windows. (Quản trị viên)
- Server.

Ứng dụng trên android cho phép người sử dụng có thể xem danh sách các công việc, chi tiết của mỗi công việc. Ứng dụng sẽ dựa trên số điện thoại để lấy dữ liệu về thông qua API truyền từ Server.

Phần mềm trên windows cho phép trưởng nhóm (quản lý - admin) có thể thêm các công việc, tạo thành các nhóm, thêm các người dùng dựa trên số điện thoại. Toàn bộ dữ liệu sẽ được lưu lên server.

Server là nơi lưu trữ toàn bộ dữ liệu. Việc cập nhật dữ liệu sẽ do admin dùng phần mềm trên windows. Khi ứng dụng android chạy server sẽ điều hành việc gửi dữ liệu xuống.

3. Sự cần thiết của ứng dụng

Với thời đại công nghệ hiện nay, việc quản lý công việc online là rất cần thiết. Chính vì thế việc ra đời các công cụ, cũng như phần mềm hỗ trợ điều này là điều tất yếu.

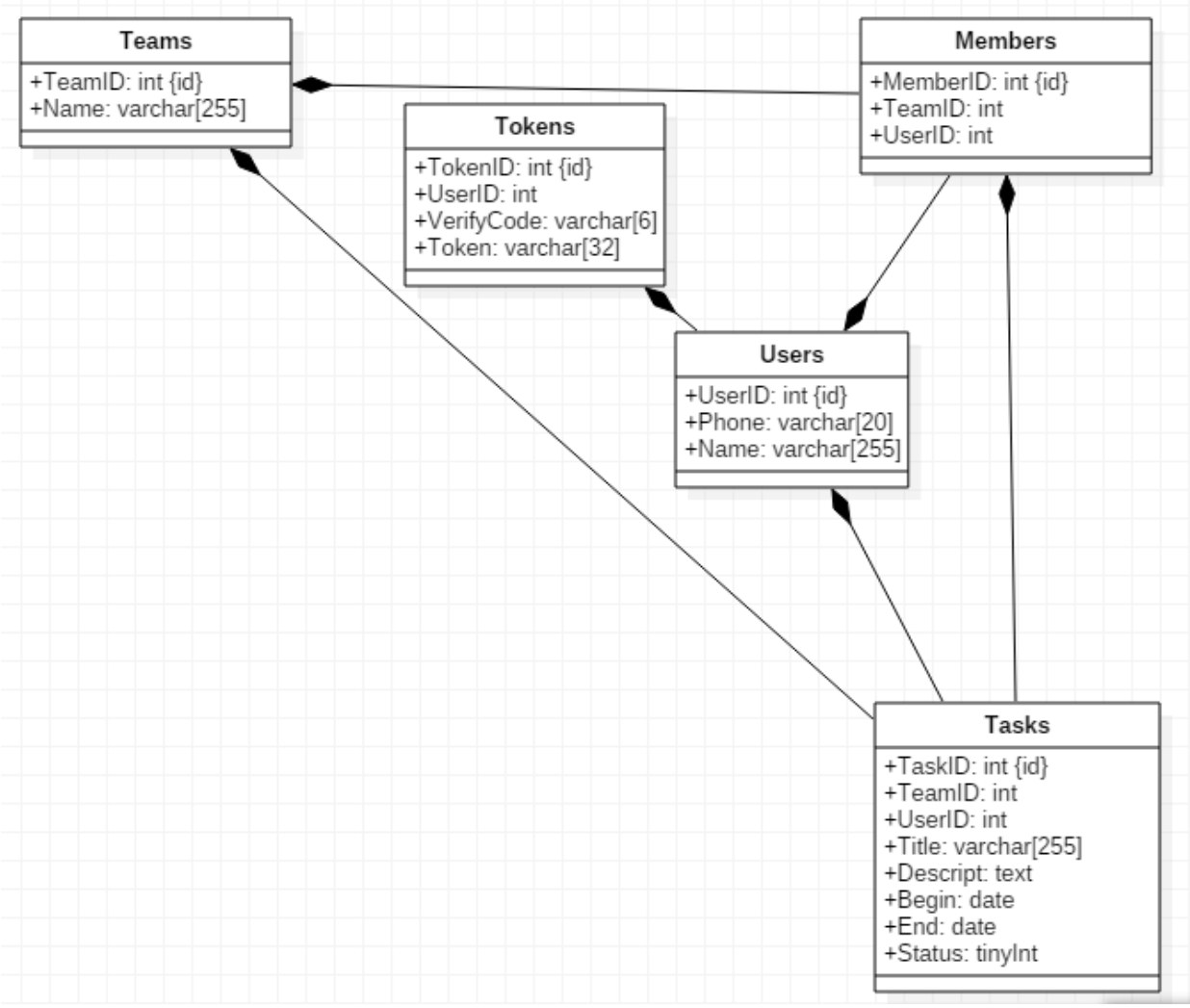
CHƯƠNG 2. PHÂN TÍCH THIẾT KẾ HỆ THỐNG

2.1 Phân tích hệ thống

Windows Application

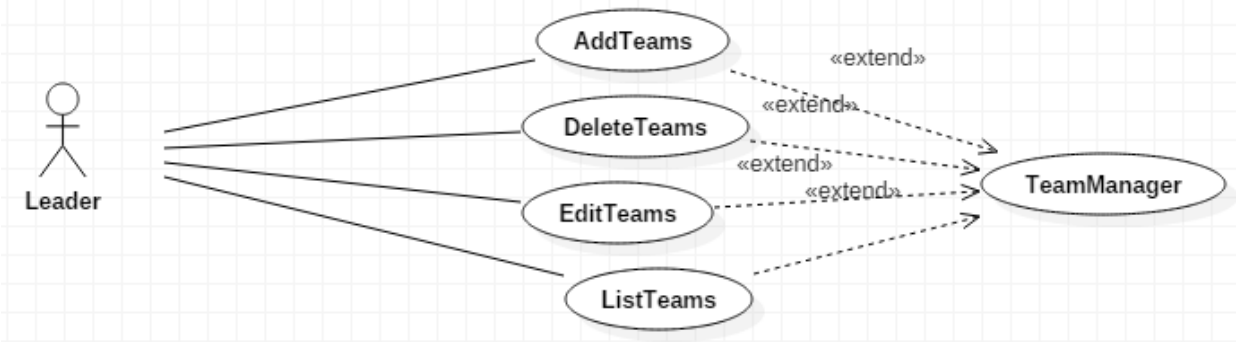
Cấu trúc Database:

Nơi sẽ lưu tất cả thông tin về: Teams, Members, Users and Tasks.

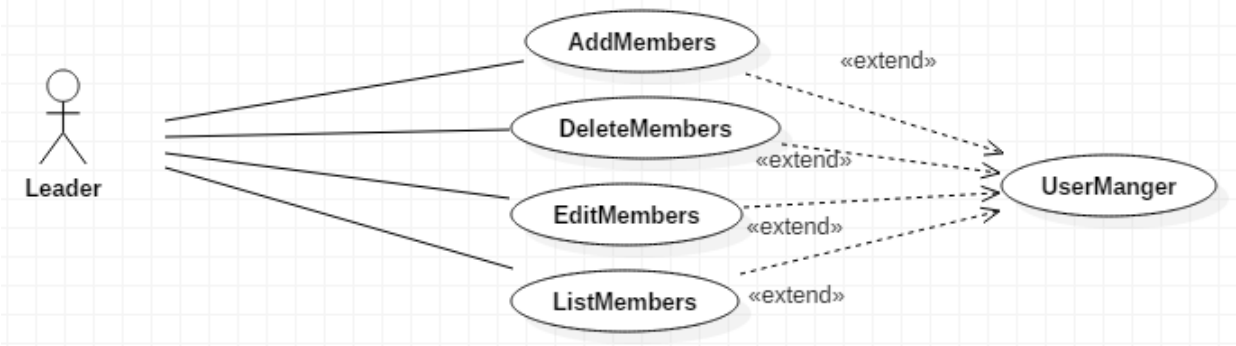


Các đặc trưng của Database

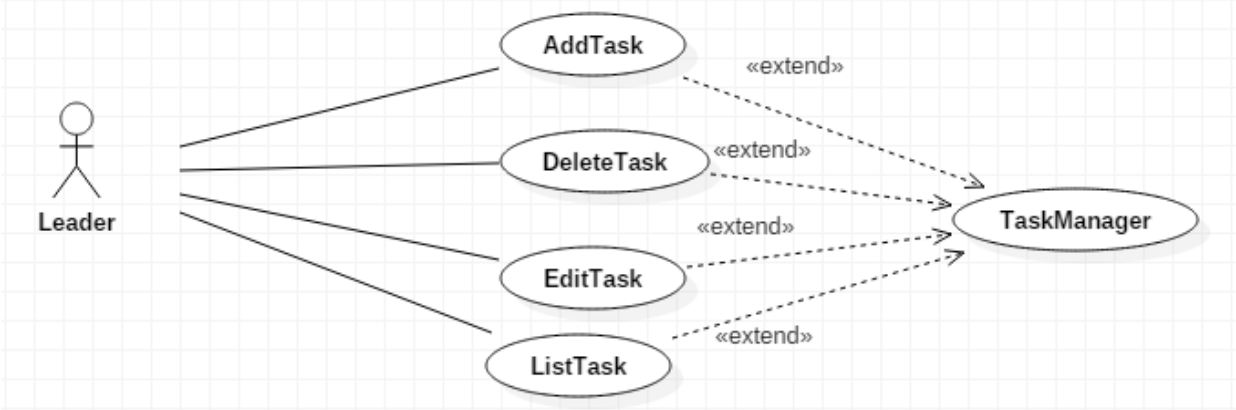
Teams : Trưởng nhóm (Leader) có thể thêm, xóa, sửa và hiển thị tất cả thành viên (Teams).Tất cả dữ liệu sẽ được quản lý bởi TeamManager.



Members: Trưởng nhóm (Leader) có thể thêm, xóa, sửa và hiển thị tất cả thành viên (Members). Tất cả dữ liệu sẽ được quản lý bởi UserManager.



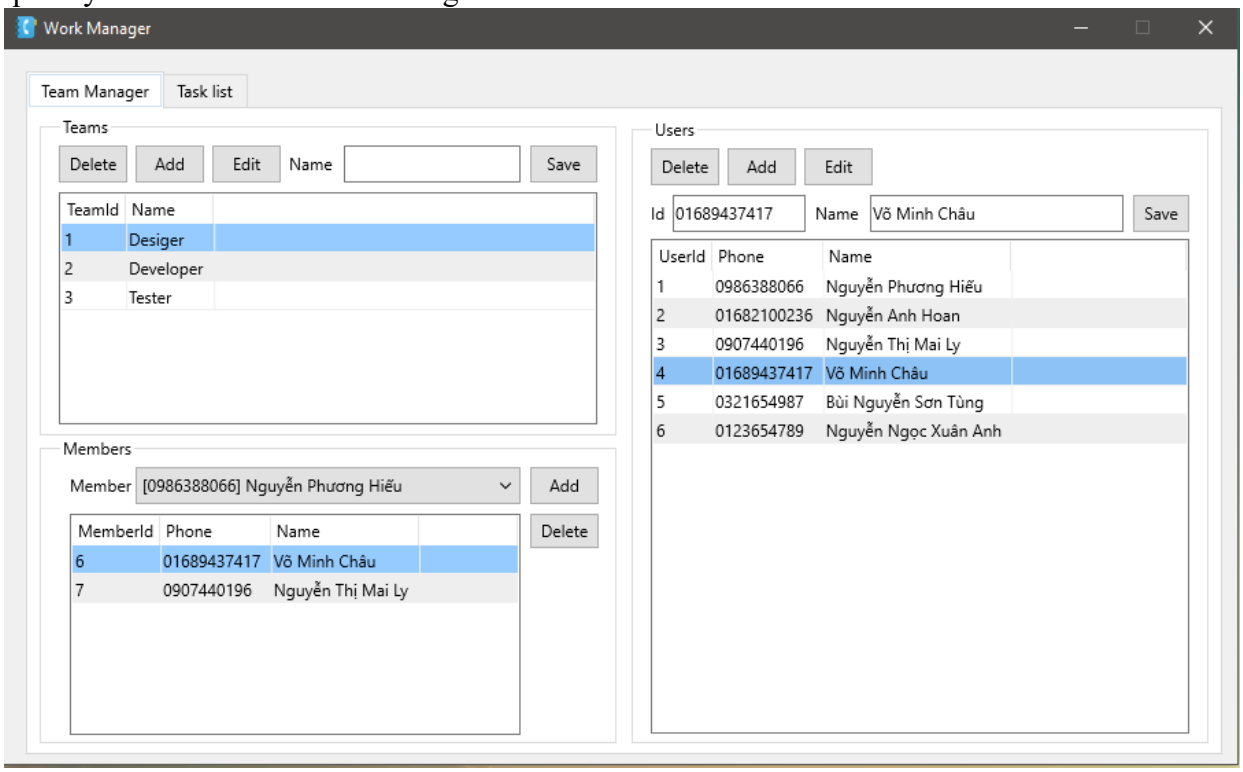
Tasks : Trưởng nhóm (Leader) có thể thêm, xóa, sửa và hiển thị tất cả công việc (Tasks). Tất cả dữ liệu sẽ được quản lý bởi TaskManager.



Manager Program

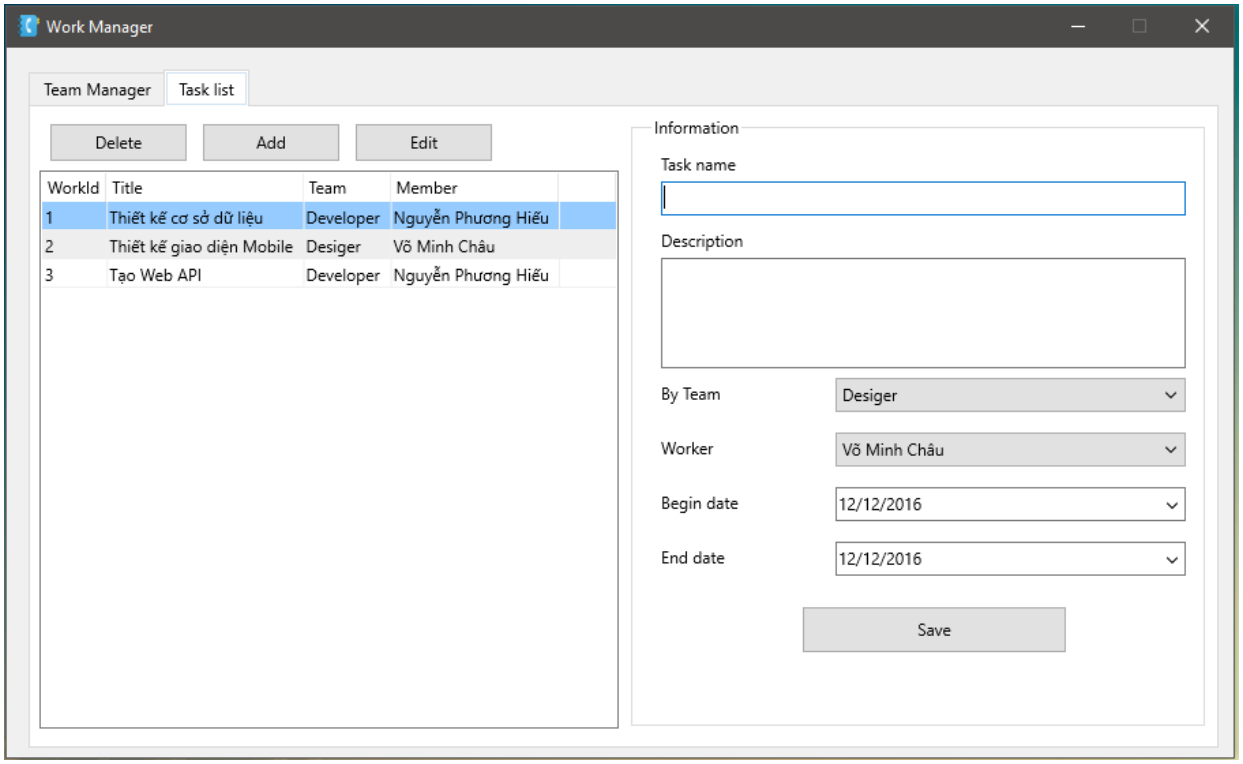
Mô tả tổng quan:

Chương trình này sẽ được cài đặt trên máy tính của Trưởng nhóm - admin để dễ quản lý. Admin có thể thêm Teams, Members, Users và Task. Mọi dữ liệu sẽ được lưu trực tiếp lên Server. Việc quản lý database sẽ trở nên dễ dàng hơn.



Team Manager

- Teams: Quản trị viên có thể thêm, xóa, sửa Teams.
- Members: Quản trị viên có thể thêm, xóa các thành viên (Members) tham gia trong Team. Danh sách thành viên (Members) sẽ được lấy từ Users. Quản trị viên có thể thêm thành viên vào Teams.
- Users: Tất cả thành viên sẽ được lấy từ server. Quản trị viên có thể thêm, xóa, sửa Members. Nó sẽ thay đổi trực tiếp đến server.



Task list

- Information: Khu vực này sẽ hiển thị tất cả thông tin từ trong Task bao gồm: Task name, Description, Description, Worker…
- Left control: Tất cả task sẽ được lấy từ server. Admin có thể thêm, xóa, sửa trong khu vực này.

Android Application

Workflow:

Tổng quan: Ứng dụng cho phép Trưởng nhóm (Leader) phân công công việc cho các thành viên dựa trên số điện thoại. Phân công, sắp xếp công việc dễ dàng và kịp thời nhắc nhở.

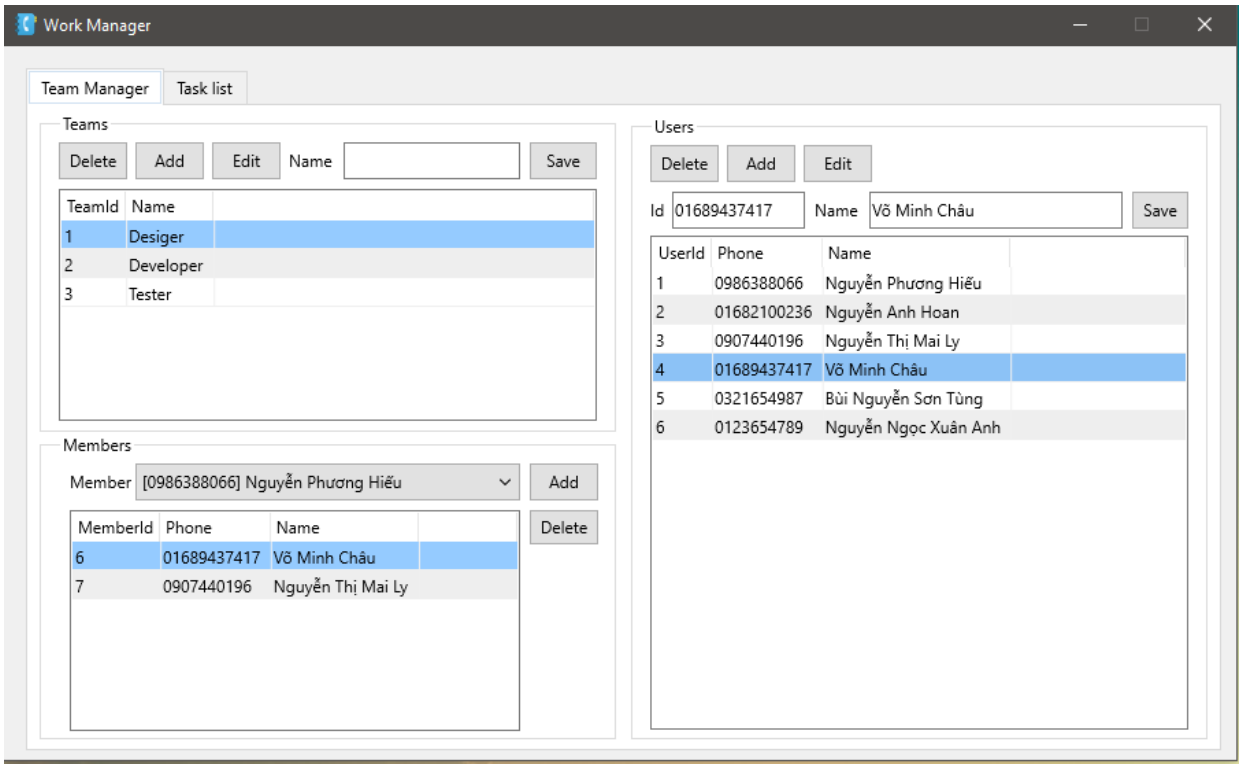
Phạm vi:

- Android OS: Android: OS 4.0.X to 4.1.X
- Các thiết bị Android dùng để kiểm thử ứng dụng: Sony Xperia Z, Asus Zenfone 5, Samsung Galaxy A3.
- Android có độ phân giải màn hình: 4 inch, 5 inch, 5.5 inch
- Chỉ hỗ trợ màn hình landscape.

Chức năng chính :

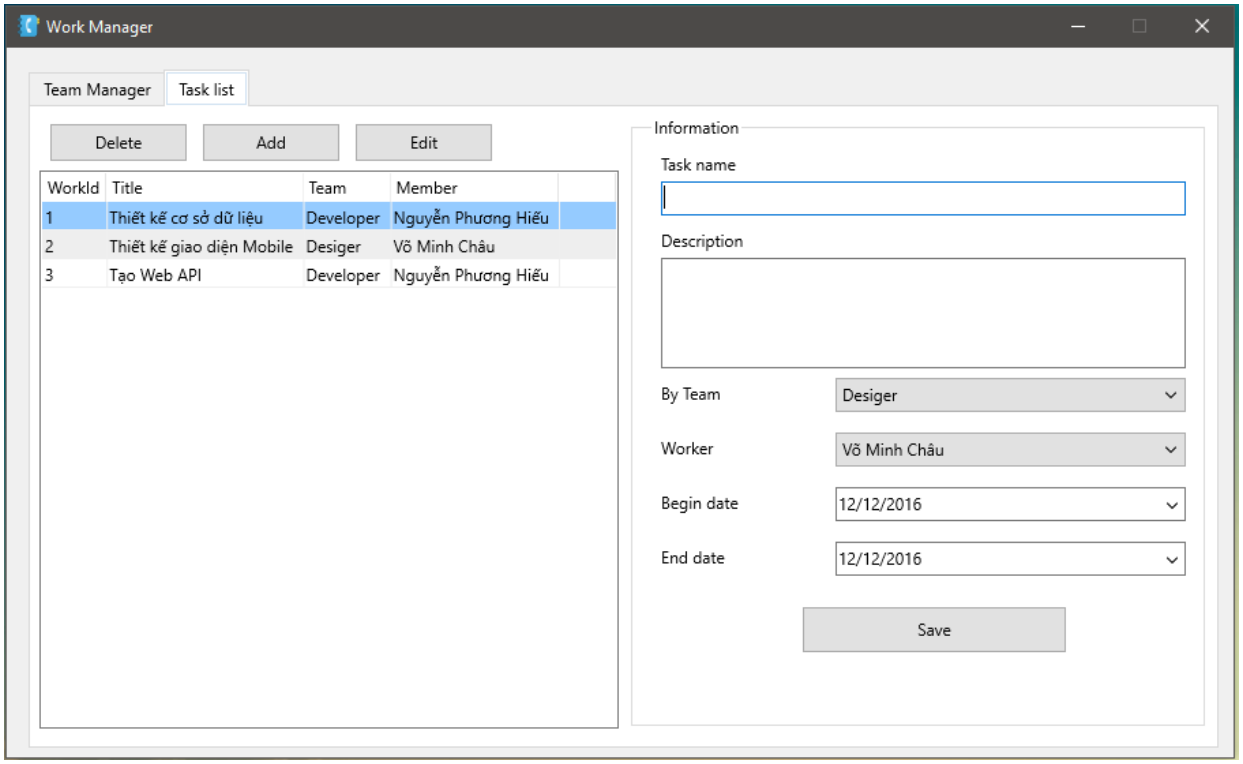
Lấy danh sách công việc từ server dựa trên số điện thoại, xem chi tiết mỗi công việc.

2.2 Thiết kế hệ thống



Team Manager
Chức năng chính : Có 3 chức năng để sử dụng :

+ Teams : Khu vực Teams cho phép ta thêm nhóm mới, sửa nhóm, xoá nhóm.
+ Member: Khi click vào 1 nhóm trong Data View của Teams, danh sách các sinh viên có trong nhóm sẽ được hiện ra ở khu vực DataView trong Member. Từ đó ta có thể chọn thêm các thành viên mới từ ô selectBox Member (các thành viên này được lấy ra từ User). Nếu muốn xoá thành viên nào ra khỏi nhóm, ta chỉ việc click vào 1 thành viên trong DataView, sau đó xoá.
+ User: ở đây ta có thể thêm, xoá sửa các thành viên. Các thành viên này sẽ được hiển thị ra DataView của Users.



Task List

Chức năng chính : Có 3 chức năng để sử dụng :

+ Information: Khu vực này sẽ hiển thị toàn bộ thông tin của Task đang được chọn, bao gồm: Task name, Description, By team, worker, begin date, end date và phím Save lưu thông tn khi sửa cũng như thêm mới.
+ Left control: Tất cả Task trên server sẽ hiển thị tại đây. Quản trị viên có thể thêm, xóa, sửa Task đang được chọn.

CHƯƠNG 3. CÀI ĐẶT VÀ KIỂM THỬ

3.1 Cài đặt

- **Login Activity**

```
public class LoginActivity extends Activity implements View.OnClickListener {
    private EditText editInput;
    private Button buttonLoginProcess;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_login);

        editInput = (EditText) findViewById(R.id.editLoginInput);
        buttonLoginProcess = (Button) findViewById(R.id.buttonLoginProcess);
        buttonLoginProcess.setOnClickListener(this);
    }

    @Override
    public void onClick(View sender) {
        switch (sender.getId())
        {
```

```
            case R.id.buttonLoginProcess:
            {
                if
(buttonLoginProcess.getText().toString().equalsIgnoreCase("LOGIN")) //Login to system
                {
                    if (editInput.getText().toString().isEmpty())
                    {
                        Toast.makeText(this, "Please enter phone number",
Toast.LENGTH_SHORT).show();
                        break;
                    }
                    LoginAPI task = new LoginAPI(this,
findViewById(R.id.activity_login) );
                    try {
                        String URLString = "https://wm.geevoo.com/login/" +
editInput.getText().toString();
                        task.execute(new URL[] { new URL(URLString) });
                    }
                    catch (Exception E) {}
                }
                else //Verify Code
                {
                    if (editInput.getText().toString().isEmpty())
                    {
                        Toast.makeText(this, "Please enter verify code",
Toast.LENGTH_SHORT).show();
                        break;
                    }
                    SharedPreferences sPreference = getSharedPreferences("UserInfo",
MODE_PRIVATE);
                    String userId = sPreference.getString("UserId", "");
                    VerifyAPI task = new VerifyAPI(this,
findViewById(R.id.activity_login) );
                    try {
                        task.execute(new URL[] { new
URL("https://wm.geevoo.com/verify/" + userId + "/" +
editInput.getText().toString().toUpperCase()) });
                    }
                    catch (Exception E) {}
                }
                break;
            }
        }
    }
}
```

- **LoginAPI**

```
public class LoginAPI extends AsyncTask<URL, Void, String> {
    private Context fContext;
    private View fView;
    public LoginAPI(Context context, View view) {
        this.fContext = context;
        this.fView = view;

        ((Button)
fView.findViewById(R.id.buttonLoginProcess)).setVisibility(View.INVISIBLE);
    }

    @Override
    protected String doInBackground(URL... urls) {
        String responseString = "";
        try
```

```
        {
            URLConnection urlConnect = urls[0].openConnection();
            BufferedReader responseStream = new BufferedReader(new
InputStreamReader(urlConnect.getInputStream()));

            StringBuilder stringBuilder = new StringBuilder();
            String lineString;
            while ((lineString = responseStream.readLine()) != null)
                stringBuilder.append(lineString + "\n");
            responseStream.close();
            responseString = stringBuilder.toString();
        }
        catch (Exception E) { }

        return responseString;
    }

    private String getUserId(String responseString)
    {
        try {
            JSONObject jObject = new JSONObject(responseString);
            return jObject.getString("userid");
        }
        catch (Exception E) { return ""; }
    }

    @Override
    protected void onPostExecute(String responseString)
    {
        String userId = getUserId(responseString);
        ((Button)
fView.findViewById(R.id.buttonLoginProcess)).setVisibility(View.VISIBLE);
        if (userId.isEmpty())
        {
            Toast.makeText(fContext, "We cannot process your request",
Toast.LENGTH_SHORT).show();
        }
        else
        {
            ((Button) fView.findViewById(R.id.buttonLoginProcess)).setText("Verify");
            ((TextView) fView.findViewById(R.id.txtLogin)).setText("Enter you code");
            ((EditText) fView.findViewById(R.id.editLoginInput)).setText("");
            ((EditText)
fView.findViewById(R.id.editLoginInput)).setInputType(InputType.TYPE_CLASS_TEXT);
            SharedPreferences sPreference = fContext.getSharedPreferences("UserInfo",
MODE_PRIVATE);
            SharedPreferences.Editor editPreference = sPreference.edit();
            editPreference.putString("UserId", userId);
            editPreference.commit();
        }
        super.onPostExecute(responseString);
    }
}
```

- **VerifyAPI**

```
public class VerifyAPI extends AsyncTask<URL, Void, String> {
    private Context fContext;
    private View fView;
    public VerifyAPI(Context context, View view) {
        this.fContext = context;
```

```
        this.fView = view;

        ((Button)
fView.findViewById(R.id.buttonLoginProcess)).setVisibility(View.INVISIBLE);
    }

    @Override
    protected String doInBackground(URL... urls) {
        String responseString = "";
        try
        {
            URLConnection urlConnect = urls[0].openConnection();
            BufferedReader responseStream = new BufferedReader(new
InputStreamReader(urlConnect.getInputStream()));

            StringBuilder stringBuilder = new StringBuilder();
            String lineString;
            while ((lineString = responseStream.readLine()) != null)
                stringBuilder.append(lineString + "\n");
            responseStream.close();
            responseString = stringBuilder.toString();
        }
        catch (Exception E) { }

        return responseString;
    }

    private String getToken(String responseString)
    {
        try {
            JSONObject jObject = new JSONObject(responseString);
            return jObject.getString("token");
        }
        catch (Exception E) { return ""; }
    }

    @Override
    protected void onPostExecute(String responseString)
    {
        if (responseString.isEmpty())
        {
            ((Button)
fView.findViewById(R.id.buttonLoginProcess)).setVisibility(View.VISIBLE);
            Toast.makeText(fContext, "We cannot process your request",
Toast.LENGTH_SHORT).show();
        }
        else
        {
            SharedPreferences sPreference = fContext.getSharedPreferences("UserInfo",
MODE_PRIVATE);
            SharedPreferences.Editor editPreference = sPreference.edit();
            String token = getToken(responseString);
            editPreference.putString("Token", token);
            editPreference.commit();
            Intent intent = new Intent(fContext, LoadingActivity.class);
            fContext.startActivity(intent);
        }
        super.onPostExecute(responseString);
    }
}
```

- **Loading Activity**

```
public class LoadingActivity extends Activity implements View.OnClickListener {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_loading);

        ((Button) findViewById(R.id.btnReload)).setOnClickListener(this);
        loadTasks();
    }
    private void loadTasks()
    {
        SharedPreferences sPreference = getSharedPreferences("UserInfo",
MODE_PRIVATE);
        String userId = sPreference.getString("UserId", "");
        String token = sPreference.getString("Token", "");
        TaskAPI task = new TaskAPI(this, findViewById(R.id.activity_loading) );
        try {
            task.execute(new URL[] { new URL("https://wm.geevoo.com/tasks/" + userId
+ "/" + token) });
        }
        catch (Exception E) {}
    }
    @Override
    public void onClick(View sender) {
        switch (sender.getId())
        {
            case R.id.btnReload:
            {
                ((Button)
findViewById(R.id.btnReload)).setVisibility(View.INVISIBLE);
                loadTasks();
                break;
            }
        }
    }
}
```

- **TaskAPI**

```
public class TaskAPI extends AsyncTask<URL, Void, String> {
    private Context fContext;
    private View fView;
    public TaskAPI(Context context, View view) {
        this.fContext = context;
        this.fView = view;

        ((ProgressBar)
fView.findViewById(R.id.prgLoading)).setVisibility(View.VISIBLE);
    }

    @Override
    protected String doInBackground(URL... urls) {
        String responseString = "";
        try
        {
            URLConnection urlConnect = urls[0].openConnection();
            BufferedReader responseStream = new BufferedReader(new
InputStreamReader(urlConnect.getInputStream()));

            StringBuilder stringBuilder = new StringBuilder();
            String lineString;
```

```
            while ((lineString = responseStream.readLine()) != null)
                stringBuilder.append(lineString + "\n");
            responseStream.close();
            responseString = stringBuilder.toString();
        }
        catch (Exception E) { }

        return responseString;
    }

    private boolean checkSuccess(String responseString)
    {
        try {
            JSONObject jObject = new JSONObject(responseString);
            return  jObject.getBoolean("status");
        }
        catch (Exception E) { return false; }
    }

    @Override
    protected void onPostExecute(String responseString)
    {
        ((ProgressBar)
fView.findViewById(R.id.prgLoading)).setVisibility(View.INVISIBLE);
        if (checkSuccess(responseString))
        {
            Intent intent = new Intent(fContext, TaskActivity.class);
            intent.putExtra("JSONData", responseString);
            fContext.startActivity(intent);
        }
        else
        {
            ((Button)
fView.findViewById(R.id.btnReload)).setVisibility(View.VISIBLE);
        }
    }
}
```

- **Task Activity**

```
public class TaskActivity extends AppCompatActivity {
    private TabLayout tabLayout;
    private ViewPager viewPager;
    private TaskListFragment taskListFragment;
    private ViewPagerAdapter adapterPager;
    private int[] tabIcons = {
            R.mipmap.task_list,
            R.mipmap.detail
    };

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

        taskListFragment = new TaskListFragment();
        taskListFragment.setArguments(getIntent().getExtras());

        adapterPager = new ViewPagerAdapter(getSupportFragmentManager());
        adapterPager.addFrag(taskListFragment, "Task List");
        adapterPager.addFrag(new DetailFragment(), "Task Detail");
```

```
        viewPager = (ViewPager) findViewById(R.id.viewpager);
        viewPager.setAdapter(adapterPager);

        tabLayout = (TabLayout) findViewById(R.id.tabs);
        tabLayout.setupWithViewPager(viewPager);
        setupTabIcons();
    }

    private void setupTabIcons() {
        tabLayout.getTabAt(0).setIcon(tabIcons[0]);
        tabLayout.getTabAt(1).setIcon(tabIcons[1]);
    }

    public void openTask(Task task)
    {
        //Update tabLayout
        ((DetailFragment) adapterPager.getItem(1)).showTask(task);
        tabLayout.setTabsFromPagerAdapter(adapterPager);
        tabLayout.getTabAt(1).select();
    }

    class ViewPagerAdapter extends FragmentPagerAdapter {
        private final List<Fragment> mFragmentList = new ArrayList<>();
        private final List<String> mFragmentTitleList = new ArrayList<>();

        public ViewPagerAdapter(FragmentManager manager) {
            super(manager);
        }

        @Override
        public Fragment getItem(int position) {
            return mFragmentList.get(position);
        }

        @Override
        public int getCount() {
            return mFragmentList.size();
        }

        public void addFrag(Fragment fragment, String title) {
            mFragmentList.add(fragment);
            mFragmentTitleList.add(title);
        }

        @Override
        public CharSequence getPageTitle(int position) {
            return mFragmentTitleList.get(position);
        }
    }
}
```

- **TaskList Fragment**

```
public class TaskListFragment extends Fragment implements View.OnClickListener,
AdapterView.OnItemClickListener {

    public TaskListFragment() {
        // Required empty public constructor
    }

    @Override
    public void onCreate(Bundle savedInstanceState) {
```

```
            super.onCreate(savedInstanceState);
        }
        private TaskManager taskManager;
        private Button btnLoad;
        @Override
        public View onCreateView(LayoutInflater inflater, ViewGroup container,
                                 Bundle savedInstanceState) {
            View view = inflater.inflate(R.layout.fragment_tasklist, container,
    false);

            taskManager = TaskManager.parseJSON(getArguments().getString("JSONData"));
            TaskAdapter adapter = new TaskAdapter(getActivity(), R.layout.task_item,
    taskManager.getTasks());

            ListView lstTask = (ListView) view.findViewById(R.id.lstTasks);
            lstTask.setOnItemClickListener(this);
            lstTask.setAdapter(adapter);

            return view;
        }

        @Override
        public void onClick(View sender) {
        }

        public void onItemClick(AdapterView<?> adapterView, View view, int i, long l)
    {
            ((TaskActivity) getActivity()).openTask(taskManager.getTasks().get(i));
        }
    }
```

- **Task Manager**

```
public class TaskManager {
    public TaskManager(boolean Status) {
        this.fStatus = Status;
        this.fTasks = new ArrayList<Task>();
    }

    public TaskManager() {
        this.fStatus = false;
        this.fTasks = new ArrayList<Task>();
    }

    public boolean getStatus() {
        return fStatus;
    }

    public void setStatus(boolean Status) {
        this.fStatus = Status;
    }

    public ArrayList<Task> getTasks() {
        return fTasks;
    }

    public void setTasks(ArrayList<Task> Tasks) {
        this.fTasks = Tasks;
    }

    private boolean fStatus;
    private ArrayList<Task> fTasks;
```

```
    public static TaskManager parseJSON(String JSONString)
    {
        TaskManager taskManager = new TaskManager();
        try
        {
            JSONObject jObject = new JSONObject(JSONString);
            taskManager.setStatus(jObject.getBoolean("status"));

            JSONArray jTasks = jObject.getJSONArray("tasks");
            for (int i = 0 ; i < jTasks.length(); i++)
            {
                Task task = new Task();
                try {

                    JSONObject jTask = jTasks.getJSONObject(i);
                    task.setId(jTask.getInt("id"));
                    task.setTitle(jTask.getString("name"));
                    task.setDescript(jTask.getString("descript"));
                    task.setTeam(jTask.getString("team"));
                    task.setBegin(jTask.getString("begin"));
                    task.setEnd(jTask.getString("end"));
                    taskManager.getTasks().add(task);
                }
                catch (Exception e) {}
            }
        }
        catch (Exception E) {}

        return taskManager;
    }
}
```

- **Task Adapter**

```
public class TaskAdapter extends ArrayAdapter<Task> {
    private Activity context = null;
    private ArrayList<Task> taskList = null;
    private int layoutId;

    public TaskAdapter(Activity context, int layoutId, ArrayList<Task> TaskList){
        super(context, layoutId, TaskList);
        this.context = context;
        this.layoutId = layoutId;
        this.taskList = TaskList;
    }

    public View getView(int position, View convertView, ViewGroup parent) {

        LayoutInflater inflater = context.getLayoutInflater();
        convertView = inflater.inflate(layoutId, null);
        if ((position >= 0) && (position < taskList.size()))
        {
            Task task = taskList.get(position);
            TextView txtTitle = (TextView) convertView.findViewById(R.id.item_title);
            txtTitle.setText(task.getTitle());
            TextView txtTeam = (TextView) convertView.findViewById(R.id.item_team);
            txtTeam.setText(task.getTeam());
        }
        return convertView;
    }
```

- **Detail Fragment**

```
public class DetailFragment extends Fragment{

    public DetailFragment() {
        // Required empty public constructor
    }

    private View fView;

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
    }

    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup container,
                             Bundle savedInstanceState) {
        fView = inflater.inflate(R.layout.fragment_detail, container, false);
        return fView;
    }

    public void showTask(Task task)
    {
        String taskInfo = "Title: \n    " + task.getTitle() + "\n";
        taskInfo += "Description: \n    " + task.getDescript() + "\n";
        taskInfo += "Team: \n    " + task.getTeam() + "\n";
        taskInfo += "Begin: \n    " + task.getBegin() + "\n";
        taskInfo += "End: \n    " + task.getEnd() + "\n";

        ((EditText) fView.findViewById(R.id.edtDetail)).setText(taskInfo);
    }
}
```

- **Task**

```
public class Task {
    private int fId;
    public Task(int Id, String Title, String Descript, String Team, String Begin,
String End) {
        this.fId = Id;
        this.fTitle = Title;
        this.fDescript = Descript;
        this.fTeam = Team;
        this.fBegin = Begin;
        this.fEnd = End;
    }

    public Task() {
        this.fId = -1;
        this.fTitle = "";
        this.fDescript = "";
        this.fTeam = "";
        this.fBegin = "";
        this.fEnd = "";
    }

    public int getId() {
        return fId;
    }
```

```
    public void setId(int fId) {
        this.fId = fId;
    }

    public String getTitle() {
        return fTitle;
    }

    public void setTitle(String fTitle) {
        this.fTitle = fTitle;
    }

    public String getDescript() {
        return fDescript;
    }

    public void setDescript(String fDescript) {
        this.fDescript = fDescript;
    }

    public String getTeam() {
        return fTeam;
    }

    public void setTeam(String fTeam) {
        this.fTeam = fTeam;
    }

    public String getBegin() {
        return fBegin;
    }

    public void setBegin(String fBegin) {
        this.fBegin = fBegin;
    }

    public String getEnd() {
        return fEnd;
    }

    public void setEnd(String fEnd) {
        this.fEnd = fEnd;
    }

    private String fTitle;
    private String fDescript;
    private String fTeam;
    private String fBegin;
    private String fEnd;

}
```

3.2 Kiểm thử

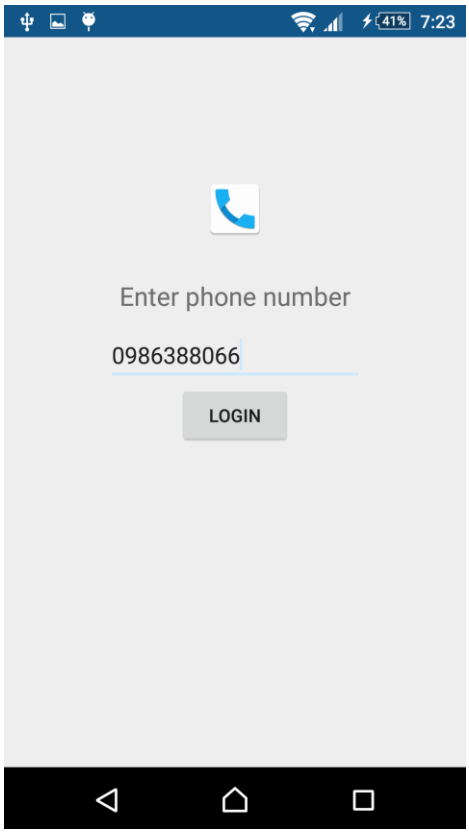
Platform / Configuration	
OS	*<OS Version>*
Devices	*Sony Xperia Z,* *Asus Zenfone 5,* *Samsung Galaxy A3.*
Screen size	*4 inch, 5 inch, 5.5 inch*

Priority	Number of Tcs
High	19
Normal	42
Low	21
Total	82

Result	Total	Build 1.0
PASSED	71	71
FAILED	11	11
BLOCKED	0	0
N/A	0	0
NOT YET TESTED	0	0
EXECUTED	82	82
PASS RATE	86,59%	86,59%

CHƯƠNG 4. KẾT QUẢ ĐẠT ĐƯỢC

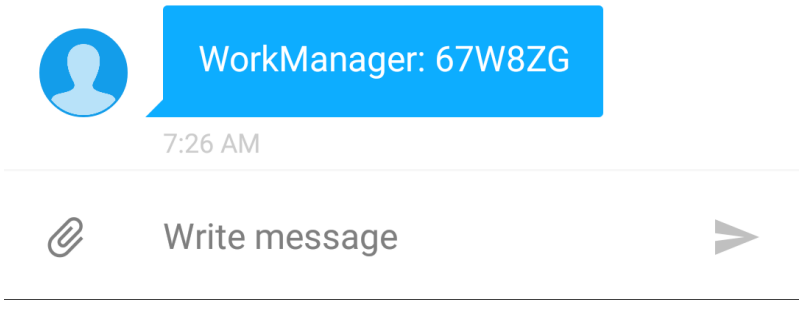
4.1 Kết quả đạt được



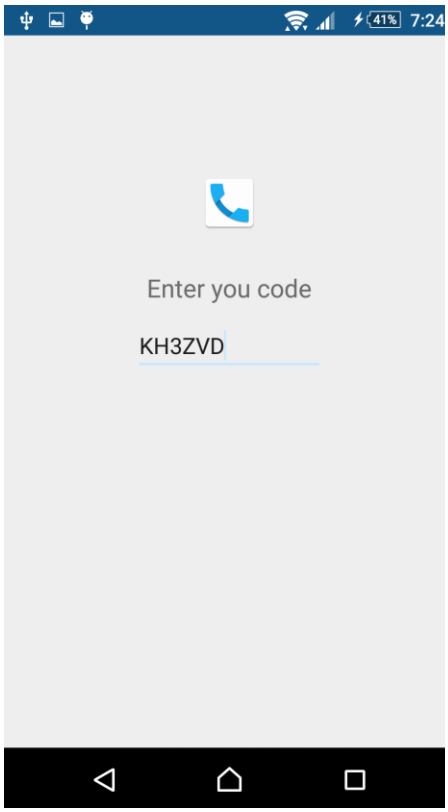
- Màn hình đăng nhập: nhập vào số điện thoại để đăng nhập

+ Nếu sdt chưa tồn tại trong hệ thống user của ứng dụng thì sẽ hiện ra popup thông báo chưa có sdt đó

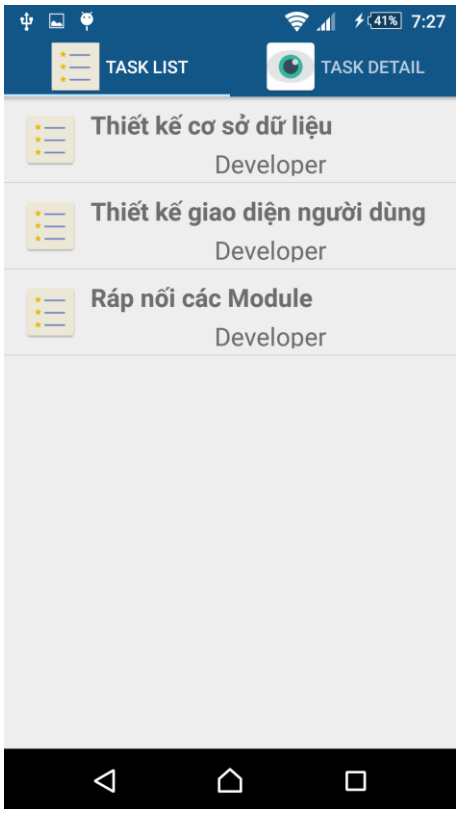
+ Nếu sdt đã có trong hệ thống, khi login server sẽ gửi về sdt 1 tin nhắn chứa mã xác nhận để đăng nhập vào



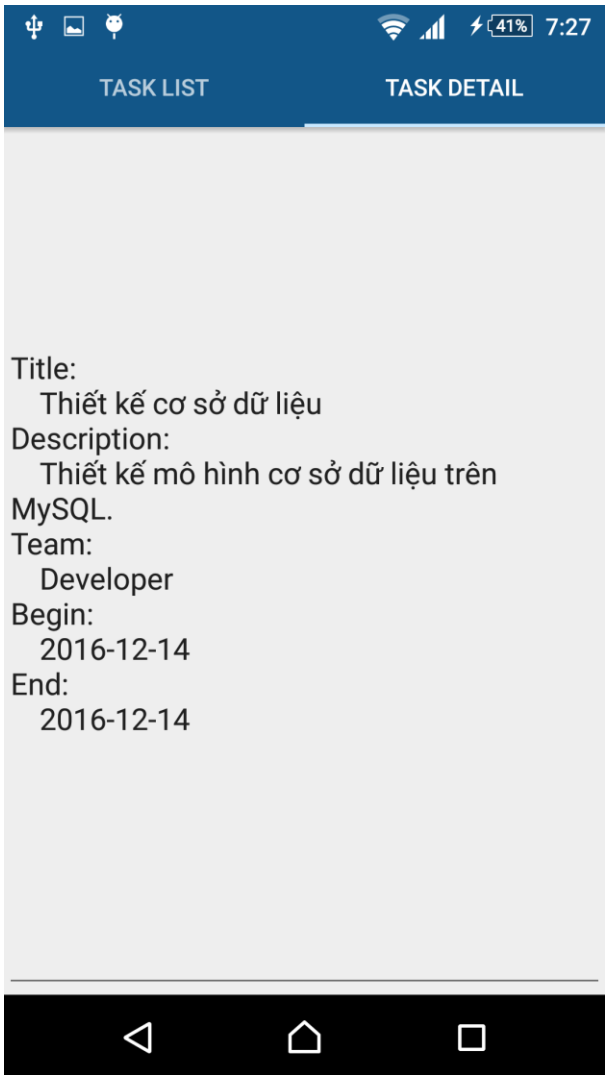
- Tin nhắn được gửi về điện thoại chứa mã xác nhận



- Sau khi nhập đúng mã xác nhận sẽ tự động đăng nhập.



- Sever sẽ load danh sách tất cả các công việc ra màn hình ở Task List. Chọn vào 1 công việc để xem chi tiết cụ thể



- Khi chọn 1 công việc, chi tiết công việc sẽ hiển thị ở tabs Task Detail

4.2 Các kết luận và kiến nghị

- Những điểm đã làm được
  +Đã lấy được thông tin công việc qua số điện thoại.
- Những điểm chưa làm được
  +Chưa lưu được thông tin người dùng.
  +Chưa xử lý được sự kiện Back.