**CHAPTER-1**

**INTRODUCTION**

**1.1 Motivation**

As a result of technological progress we are facing an incredible variety of possibilities to communicate regardless of the distance. Smart phones provide a great choice of features that facilitate the life for the users as well they make it more comfortable.

Every day the features and capabilities of Mobiles are increasing surprisingly. For this reason we want to create an Android application which gives us the opportunity to improve our knowledge of Mobile developing. We find this project a great opportunity to combine a lot of technologies and languages in the same software system, and learn how to work in big projects as a team member. An important motivation for us is the challenge to solve all the problems that will arise while the project is developed. In addition, we will try to make the program portable, reliable, secure, stable, intuitive, etc.

**1.2 Problem Definition**

This application will allow the citizens to send the complaints directly from their smart phone and enable the citizens of city to lodge complaints anytime, anywhere.

This Application will help the common people under the jurisdiction of a municipal corporation to register their complaints about day to day problems in their ward through a mobile application. It will provide a common man to deliver his complaints and problems to municipal authority as well as let the municipal authorities to address the problem in a short period of time. This application provides an interface to register one’s complained and follow it up.

**1.3 Objective of Project**

The main objective of this project is to provide the citizens to send the complaints directly from their smart phone and enable the citizens of city to lodge complaints anytime, anywhere. This application will help the municipal authorities to address the problem in a very short period of time. The main purpose of this project is to help the public in knowing their place details and getting their problems solved in online without going to the officer regularly until the problem is solved. By this system the public can save his time and eradicate corruption in government offices. Its main purpose is to provide a smart and easy way through Android Application for Complaint registration.

**1.4. Limitations of Project**

The limitations of project are:

* We can’t upload the images related to the problem.

**CHAPTER**-**2**

**LITERATURE SURVEY**

**2.1 Introduction**

Android applications have become the most important public communication medium for every individual. In India we don’t have any direct communication between the government and public in an efficient way for solving the problems i.e. for getting a problem solved in our place we have to bribe the officials and get them solved in 2 months which can be solved actually in 1 month of time. Initially phones were merely used for calling or texting. Now-a-days, the scenario has changed. In today’s world, more focus is given on the availability of the internet and thus using various applications present in the android market. Now days we are supposed to manage our daily work on time, accurately, very fast and with our satisfaction. So we are using various technologies in our life for fulfillment of our daily work. The main purpose of this project is to help the public in knowing their place details and getting their problems solved in online without going to the officer regularly until the problem is solved. By this system the public can save his time.

**2.2 Existing System**

**2.2.1Through Web Portal**

* The web portal while allowing the users to lodge complaint has some serious constrains.
* The user is restricted to choose from among the list of predefined drop down list. This poses a major problem in the sense that the user tries to fit his complaint intone of the predefined drop downs. This results in a wrong fit and the complaint being directed to the wrong department which in turn affects the time taken to resolve the problem.
* The second major constraint is the need for the user to be aware of the ward number, additionally the user needs to have access to networked computer.

**2.2.2 Prajavani:**

* Prajavani is an e-governance initiative by the combined efforts of District administration and National Informatics center in Ranga Reddy. Prajavani is a unique public private partnership program, which gives citizens an opportunity to interact with government without coming to any government office.
* No registration facility is provided and can post the problems directly.
* Users can post their problems but cannot get the details regarding progress of their complaint.

**2.3. Disadvantages of Existing System**

* No registration facility is provided and can post the problems directly.
* Users can post their problems but cannot get the details regarding progress of their complaint.

**2.4 Proposed System**

* In the proposed system the citizen need not go to the government office for getting his problem solved. The proposed system will enable the citizens of city to lodge complaints anytime, anywhere.
* The users use the mobile phone and do not need to access the web portal interface directly to file their complaint. The user runs the Application and can lodge complaint. Once all the information about complaint is entered by the user the details are stored in the data base.

**2.5 Conclusion**

As mobile use continues to grow worldwide, the “app vs web” question will remain a very real consideration for organizations seeking to establish a mobile presence.

Due to the following features we go with the mobile application:

Applications are usually 1.5 times faster than mobile websites and they perform actions much faster too. Applications store their data locally on your device. Allow, data retrieval happens in the blink of an eye. In the case of mobile-sites, data needs to be fetched from the web servers which can take from few seconds to a minute depending upon the network speed and packet sizes. Another reason is, mobile-sites use JavaScript to run functions and apps run on frameworks which can be five times faster. As all of this happens in the back-end, your users get to perform actions much faster on the front-end delighting them with a seamless experience.

**CHAPTER-3**

**ANALYSIS**

**3.1 Introduction**

Android is a mobile operating system developed by Google, based on a modified version of the Linux kernel and other open source software and designed primarily for touch screen mobile devices such as smart phones and tablets.

Since Android devices are usually battery-powered, Android is designed to manage processes to keep power consumption at a minimum. When an application is not in use the system suspends its operation so that, while available for immediate use rather than closed, it does not use battery power or CPU resources. Android manages the applications stored in memory automatically: when memory is low, the system will begin invisibly and automatically closing inactive processes, starting with those that have been inactive for the longest amount of time.

Android devices incorporate many optional hardware components, including video cameras, GPS, orientation sensors, dedicated gaming controls, accelerometers, gyroscopes, barometers, magnetometers, proximity sensors, thermometers and touch screens. Some hardware components are not required, but became standard in certain classes of devices, such as smart phones, and additional requirements apply if they are present. Some other hardware was initially required, but those requirements have been relaxed or eliminated altogether. For example, as Android was developed initially as a phone OS, hardware such as microphones were required, while over time the phone function became optional. Android used to require an autofocus camera, which was relaxed to a fixed-focus camera if present at all, since the camera was dropped as a requirement entirely when Android started to be used on set-top boxes.

Android's source code is released by Google under an open source license, and its open nature has encouraged a large community of developers and enthusiasts to use the open-source code as a foundation for community-driven projects, which deliver updates to older devices, add new features for advanced users or bring Android to devices originally shipped with other operating systems. These community-developed releases often bring new features and updates to devices faster than through the official manufacturer/carrier channels, with a comparable level of quality, provide continued support for older devices that no longer receive official updates; or bring Android to devices that were officially released running other operating systems, such as the HP Touchpad.

**3.1.1 Features of Android:**

Android is a powerful operating system competing with Apple 4GS and supporters great features. Few of them are listed below:

Table 3.1: FEATURES OF ANDROID

|  |  |
| --- | --- |
| Feature | Description |
| BeautifulUI  Connectivity | Android OS basic screen provides a beautiful and intuitive user interface  GSM/EDGE, IDEN, CDMA, EV-DO, UMTS, Bluetooth, Wi-Fi, LTE, NFC and Wi MAX. |
| Storage | SQLite, a lightweight relational database, is used for data storage purposes. |
| Media Support | H.263, H.264, MPEG-4 SP, AMR, AMR-WB, AAC, HE-AAC, AAC 5.1, MP3, MIDI, OggVorbis, WAV, JPEG, PNG, GIF, and BMP |
| Messaging | SMS and MMS |
| Web Browser | Based on the open-source Web Kit layout engine, coupled with Chrome's V8 JavaScript engine supporting HTML5 and CSS3. |
| Multi touch | Android has native support for multi-touch which was initially made available in handsets such as the HTCHero. |
| Multi tasking | User can jump from one task to another and same time variousapplication can run simultaneously. |
| Resizable widgets | Widgets are resizable, so users can expand them to show more content or shrink them to save space. |
| Multi-Language | Supports single direction and bi-directional text. |
| GCM | Google Cloud Messaging (GCM) is a service that lets developers send short message data to their users on Android devices, without needing a proprietary sync solution. |
| Wi-Fi Direct | A technology that lets apps discover and pair directly, over a high-bandwidth peer-to-peer connection. |
| Android Beam | A popular NFC-based technology that lets users instantly share, just by touching two NFC enabled phones together. |

**3.2 Software Requirements Specifications:**

Software Requirement Specification (SRS) is the starting point of the software development activity. It is a complete description of the behavior of a system which is to be developed. The SRS document enlists all necessary requirements for project development. To derive the requirements we need to have clear and thorough understanding of the product which is to be developed. This is prepared after detailed communication with project team and the customer.

A SRS is a comprehensive description of the intended purpose and environment for software under development. The SRS fully describes what the software will do and how it will be expected to perform.

An SRS minimizes the time and effort required by developers to achieve desired goals and also minimizes the development cost. A good SRS defines how an application will interact with system hardware, other programs and human users in a wide variety of real-world situations.

**Characteristics of SRS:**

**Correct** - An SRS is correct if, and only if, every requirement stated therein is one that the software shall meet. Traceability makes this procedure easier and less prone to error. **Unambiguous** - An SRS is unambiguous if, and only if, every requirement stated therein has only one interpretation. As a minimum, this requires that each characteristic of the final product be described using a single unique term.

**Verifiable** – It is verifiable if there exists some finite cost-effective process with which a person or machine check whether software product meets requirements.

**Consistent** - Consistency refers to internal consistency. If an SRS does not agree with some higher-level document, such as a system requirements specification, then it is not correct. An SRS is internally consistent if, and only if, no subset of individual requirements described in it conflict.

**Modifiable** – SRS is said to be modifiable if its structure and style are such that any changes to the requirements can be made easily, completely and consistently while retaining the structure and style.

**Traceable** – SRS is said to be traceable if the origin of each of its requirements is clear and it facilitates the referencing of each requirement in future enhancement.

**Ranked for importance or stability** – SRS is ranked for importance or stability if each requirement in it has an identifier to indicate either the importance or stability of that particular requirement.

**3.2.1 User Requirements:**

**3.2.1.1 Android Studio:**

Android Studio is the official Integrated Development Environment (IDE) for Android app development, based on IntelliJ IDEA. On top of IntelliJ's powerful code editor and developer tools, Android Studio offers even more features that enhance your productivity when building Android apps, such as:

* A flexible Gradle-based build system
* A fast and feature-rich emulator
* A unified environment where you can develop for all Android devices
* Instant Run to push changes to your running app without building a new APK
* Code templates and GitHub integration to help you build common app features and import sample code.
* Extensive testing tools and frameworks.
* Lint tools to catch performance, usability, version compatibility, and other problems
* C++ and NDK support
* Built-in support for Google Cloud Platform, making it easy to integrate Google Cloud
* Messaging and App Engine.

**Project Structure:**

Each project in Android Studio contains one or more modules with source code files and resource files. Types of modules include:

* Android app modules
* Library modules
* Google App Engine modules

By default, Android Studio displays your project files in the Android project view, as shown in figure 1. This view is organized by modules to provide quick access to your project's key source files.

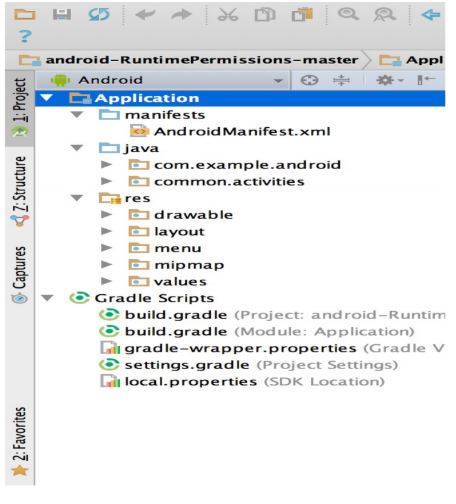


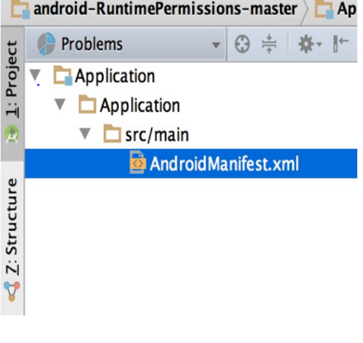
Figure 3.2.1.1: Project Files in Android View

All the build files are visible at the top level under Gradle Scripts and each app module contains the following folders:

* Manifest: Contains the AndroidManifest.xml file.
* Java: Contains the Java source code files, including JUnit test code.
* Res: Contains all non-code resources, such as XML layouts, UI strings, and bitmap images.

The Android project structure on disk differs from this flattened representation. To see the actual file structure of the project, select Project from the Project dropdown (in figure 1, it's showing as Android).

You can also customize the view of the project files to focus on specific aspects of your app development. For example, selecting the Problems view of your project displays links to the source files containing any recognized coding and syntax errors, such as a missing XML element closing tag in a layout file.

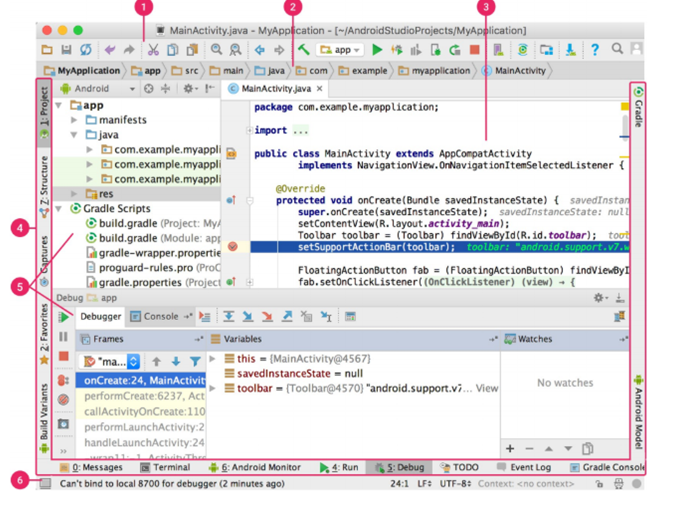


3.2.1.2 The project files in problem view

**The User Interface:**

The Android Studio main window is made up of several logical areas identified in figure 3.

* The toolbar lets you carry out a wide range of actions, including running your app andlaunching Android tools.
* The navigation bar helps you navigate through your project and open files for editing.
* It provides a more compact view of the structure visible in the Project window.



3.2.1.3 Android Studio main window

* The tool window bar runs around the outside of the IDE window and contains the buttons that allow you to expand or collapse individual tool windows.
* The tool windows give you access to specific tasks like project management, search version control, and more. You can expand them and collapse them.
* The status bar displays the status of your project and the IDE itself, as well as any warnings or messages.

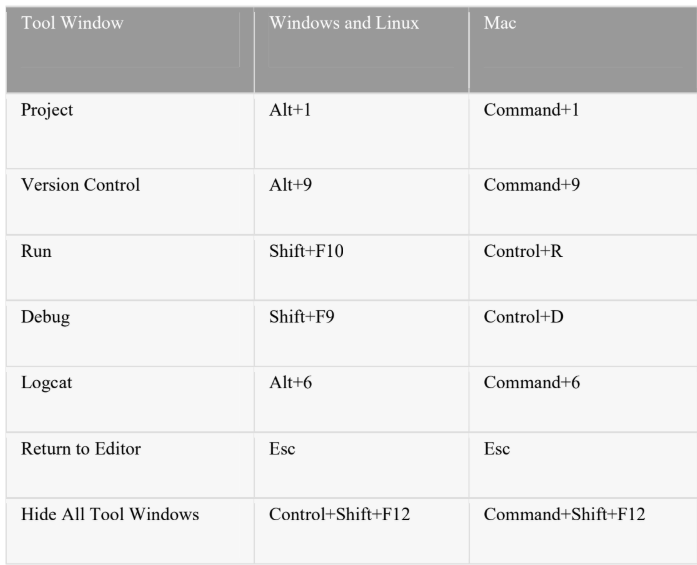
You can also customize the view of the project files to focus on specific aspects of your app development. For example, selecting the Problems view of your project displays links to the source files containing any recognized coding and syntax errors, such as a missing XML element closing tag in a layout file.

You can organize the main window to give yourself more screen space by hiding or moving toolbars and tool windows. You can also use keyboard shortcuts to access most IDE features.

At any time, you can search across your source code, database`s, actions, elements of the user interface, and so on, by double-pressing the Shift key, or clicking the magnifying glass in the upper right-hand corner of the Android Studio window. This can be very useful if, for example, you are trying to locate a particular IDE action that you have forgotten how to trigger.

Tool Windows Instead of using present perspectives, Android Studio follows your context and automatically brings up relevant tool windows as you work. By default, the most commonly used tool windows are pinned to the tool window bar at the edges of the application window.

Table 3.2: KEYBOARD SHORTCUTS FOR SOME USEFUL TOOL WINDOWS



If you want to hide all toolbars, tool windows, and editor tabs, click View > Enter Distraction Free Mode. This enables Distraction Free Mode. To exit Distraction Free Mode, click View > Exit Distraction Free Mode.

You can use Speed Search to search and filter within most tool windows in Android Studio. To use Speed Search, select the tool window and then type your search query.

**Code Completion:**

Android Studio has three types of code completion, which you can access using keyboard shortcuts.

**Find sample code:**

The Code Sample Browser in Android Studio helps you find high-quality, Google provided Android code samples based on the currently highlighted symbol in your project.For more information, see Find Sample Code.

**Navigation:**

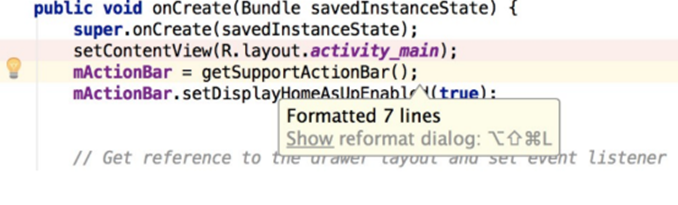
Here are some tips to help you move around Android Studio.

* Switch between your recently accessed files using the Recent Files action. Press Control+E (Command+E on a Mac) to bring up the Recent Files action.
* By default, the last accessed file is selected. You can also access any tool window through the left column in this action.
* View the structure of the current file using the File Structure action. Bring up the File Structure action by pressing Control+F12 (Command+F12 on a Mac). Using this action, you can quickly navigate to any part of your current file.
* Search for and navigate to a specific class in your project using the Navigate to Class action. Bring up the action by pressing Control+N (Command+O on a Mac).
* Navigate to Class supports sophisticated expressions, including camel humps, paths, line navigate to, middle name matching, and many more. If you call it twice in a row, it shows you the results out of the project classes.
* Navigate to a file or folder using the Navigate to file action. Bring up the Navigate to file action by pressing Control+Shift+N (Command+Shift+O on a Mac). To search for folders rather than files, add a / at the end of your expression.
* Navigate to a method or field by name using the Navigate to Symbol action. Bring up the Navigate to Symbol action by pressing Control+Shift+Alt+N (Command+Option+O on a Mac).
* Find all the pieces of code referencing the class, method, field, parameter, or statement at the current cursor position by pressing Alt+F7 (Option+F7on a Mac).

**Style and Formatting**

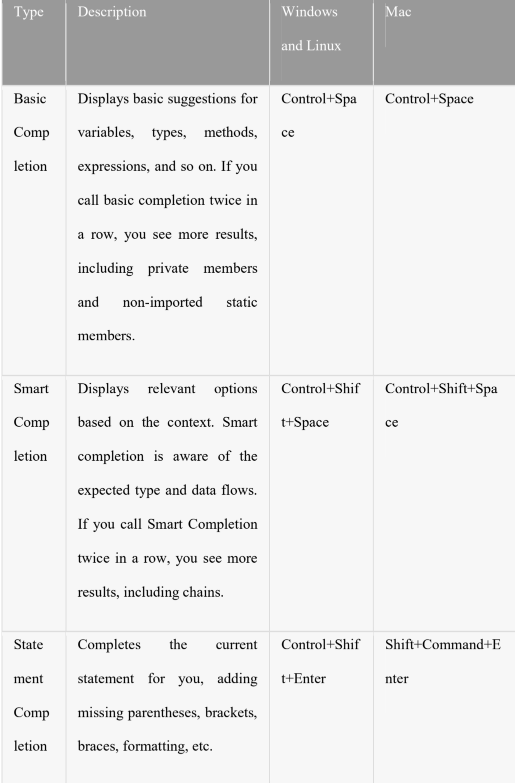
You can customize the code style settings by programming language, including specifying conventions for tabs and indents, spaces, wrapping and braces, and blank lines. To customize your code style settings, click File> Settings > Editor > Code Style (Android Studio > Preferences > Editor > Code Style on a Mac.)

Although the IDE automatically applies formatting as you work, you can also explicitly call the Reformat Code action by pressing Control+Alt+L(Opt+Command+L on a Mac), or auto-indent all lines by pressing Control+Alt+I (Control+Option+I on a Mac).



3.2.1.4 Code after formatting

Table 3.3: KEYBOARD SHORTCUTS FOR CODE COMPLETION



**Version Control Basics**

Android Studio supports a variety of version control systems (VCS’s), including GitHub, CVS, Mercurial, Subversion, and Google Cloud Source Repositories.

After importing your app into Android Studio, use the Android Studio VCS menu options to enable VCS support for the desired version control system, create a repository, import the new files into version control, and perform other version control operations:

* From the Android Studio VCS menu, click Enable Version Control Integration.
* From the drop-down menu, select a version control system to associate with the project root, and then click OK.
* The VCS menu now displays a number of version control options based on the system you selected.

**3.2.1.2 Gradle Build System:**

Android Studio uses Gradle as the foundation of the build system, with more Android-specific capabilities provided by the Android plugin for Gradle.

This build system runs as an integrated tool from the Android Studio menu, andindependently from the command line. You can use the features of the build system to do thefollowing:

* Customize, configure, and extend the build process.
* Create multiple APKs for your app, with different features using the same project and modules.
* Reuse code and resources across source sets.

By employing the flexibility of Gradle, you can achieve all of this without modifying your app's core source files.

Android Studio build files are named build.gradle. They are plain text files that use Groovy syntax to configure the build with elements provided by the Android plugin for Gradle. Each project has one top-level build file for the entire project and separate module-level build files for each module. When you import an existing project, Android Studio automatically generates the necessary build files.

**Build Variants**

The build system can help you create different versions of the same application from a single project. This is useful when you have both a free version and a paid version of your app, or if you want to distribute multiple APKs for different device configurations on Google Play.

**Multiple APK Support**

Multiple APK support allows you to efficiently create multiple APKs based on screen density or ABI. For example, you can create separate APKs of an app for the hdpi and mdpi screen densities, while still considering them a single variant and allowing them to share test APK, javac, dx, and ProGuard settings.

**Resource Shrinking**

Resource shrinking in Android Studio automatically removes unused resources fromyour packaged app and library dependencies. For example, if your application isusing Google Play services to access Google Drive functionality and you are not currentlyusing Google Sign-In, then resource shrinking can remove the various drawable assets forthe SignIn Button buttons.

**3.2.1.3 Managing Dependencies:**

Dependencies for your project are specified by name in the build gradle file. Gradletakes care of finding your dependencies and making them available in your build. You candeclare module dependencies, remote binary dependencies, and local binary dependencies inyour build.gradle file. Android Studio configures projects to use the Maven CentralRepository by default. (This configuration is included in the top-level build file for theproject.) For more information about configuring dependencies, read Add BuildDependencies.

**Debug and Profile Tools**

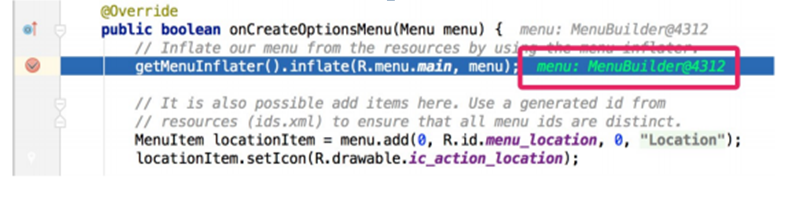
Android Studio assists you in debugging and improving the performance of your code, including inline debugging and performance analysis tools.

**Inline debugging**

Use inline debugging to enhance your code walk-through in the debugger view withinline verification of references, expressions, and variable values. Inline debug informationincludes:

**Inline variable values**

* Referring objects that reference a selected object
* Method return values
* Lambda and operator expressions
* Tooltip values



3.2.1.5 An inline variable value

**Performance profilers**

Android Studio provides performance profilers so you can more easily track your app’s memory and CPU usage, find deallocated objects, locate memory leaks, optimize graphics performance, and analyse network requests. With your app running on a device or emulator, open the Android Profile tab.

**Heap dump**

When you’re profiling memory usage in Android Studio, you can simultaneously initiate garbage collection and dump the Java heap to a heap snapshot in an Android-specific HPROF binary format file. The HPROF viewer displays classes, instances of each class, and a reference tree to help you track memory usage and find memory leaks.

**Memory Profiler**

You can use Memory Profiler to track memory allocation and watch where objects are being allocated when you perform certain actions. Knowing these allocations enables you to optimize your app’s performance and memory use by adjusting the method calls related to those actions.

**Data file access**

The Android SDK tools, such as Systrace, and logcat, generate performance and debugging data for detailed app analysis. To view the available generated data files, open the Captures tool window. In the list of the generated files, double-click a file to view the data. Right-click any .hprof files to convert them to the standard Investigate Your RAM usage file format.

**Code inspections**

Whenever you compile your program, Android Studio automatically runs configured Lint and other IDE inspections to help you easily identify and correct problems with the structural quality of your code. The Lint tool checks your Android project source files for potential bugs and optimization improvements for correctness, security, performance, usability, accessibility, and internationalization.

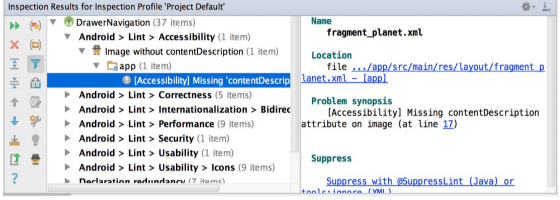
****

Figure 3.2.1.6: The results of a Lint inspection in Android Studio

In addition to Lint checks, Android Studio also performs IntelliJ code inspections and validates annotations to streamline your coding workflow.

**Annotations in Android Studio:**

Android Studio supports annotations for variables, parameters, and return values to help you catch bugs, such as null pointer exceptions and resource type conflicts. The Android SDK Manager packages the Support-Annotations library in the Android Support Repository for use with Android Studio. Android Studio validates the configured annotations during code inspection.

**3.2.2 Software Requirement**

* Operating System: Windows 7(and above).
* Coding Language: JAVA
* Screen Resolution :1280 x 800 minimum screen resolution
* Tools: Android Studio, JDK 1.8
* Data Base: SQLite

**3.2.3 Hardware Requirement**

* System : inteli3 (and above).
* Hard Disk : 500 GB.
* Ram : 4 GB.

**3.3 Flow Chart**

Create account/Sign in

Enter Username and password

No

If Valid

Add Complaints

Get Status

**Algorithm:**

Step1: start

Step2: Create account/sign in

Step3: Read username and password

Step4: if valid

Add Complaints, View status

else

Goto step2

Step5: stop

**3.4 Android installation Procedure**

Setting up Android development environment takes some time at first. It helps to make sure you don’t do anything wrong to save yourself from the agony of doing the whole process again.

You're required to have [Windows XP](http://searchenterprisedesktop.techtarget.com/definition/Windows-XP) or later, or Mac OS X 10.5.8 or a later version to start Android application development process. Then, there are four tools that you will need and they are available on the Internet for free:

1. Java JDK5 or JDK6

2. Android SDK

**Step 1: Setup Java Development Kit (JDK)**

You can download [the JDK](http://www.oracle.com/technetwork/java/javase/downloads/index.html) and install it, which is pretty easy. After that, you just have to set PATH and JAVA\_HOME variables to the folder where you have **java**and **javac.**

**Note for Windows Users:**If you installed the JDK in C:\jdk1.6.0\_15 then you will have to add the following two lines in your command prompt

C:\autoexec.bat file.

SetPATH=C:\jdk1.6.0\_15\bin;%PATH%  
set JAVA\_HOME=C:\jdk1.6.0\_15

**Step 2: Downloading and setting up Android Studio**

Google provides Android Studio for the Windows, Mac OS X, and Linux platforms. You can [download this software](https://developer.android.com/studio/index.html) from the Android Studio homepage. (You'll also find the traditional SDKs, with Android Studio's command-line tools, available from the Downloads page.)

**Installing Android studio:**

I launched android-studio-bundle-143.2821654-windows.exe to start the installation process. The installer responded by presenting the Android Studio Setup dialog box shown below.



Fig.3.4.1. Set up Android Studio

Clicking Next took me to the following dialog box, which gives you the option to decline installing the Android SDK (included with the installer) and an AVD.

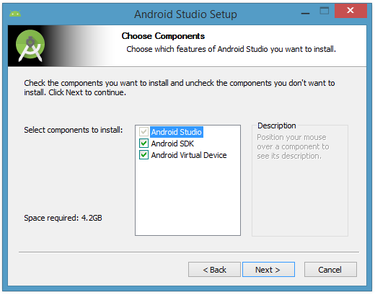


Fig.3.4.2. Do you want to install the Android SDK and AVD

I choose to keep the default settings. After clicking next, you'll be taken to the license agreement dialog box. Accept the license to continue the installation.

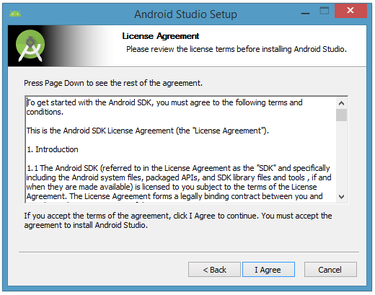


Fig.3.4.3 Accept the license agreement to continue installation

The next dialog box invites you to change the installation locations for Android Studio and the Android SDK.

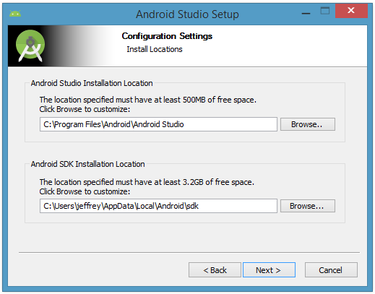


Fig.3.4.4. Set the Android Studio and Android SDK installation locations

Change the location or accept the default locations and click Next. The installer defaults to creating a shortcut for launching this proDAgram, or you can choose to decline. I recommend that you create the shortcut, and then click the Install button to begin installation.

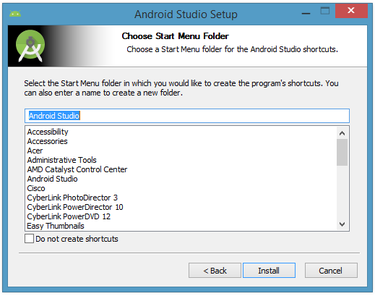


Fig.3.4.5. Create a new shortcut for Android Studio

The resulting dialog box shows the progress of installing Android Studio and the Android SDK. Clicking the Show Details button will let you view detailed information about the installation progress.

The dialog box will inform you when installation has finished. When you click next, you should see the following:

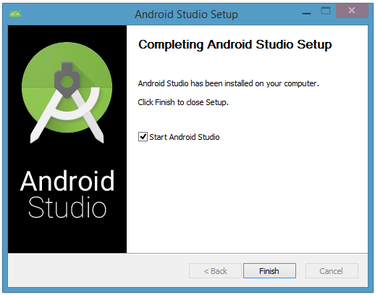


Fig.3.4.6. Leave the Start Android Studio check box checked to run this software.

To complete your installation, leave the Start Android Studio box checked and click Finish.

## Running Android Studio

Android Studio presents a splash screen when it starts running:



Fig.3.4.7. Android Studio's start screen

On your first run, you'll be asked to respond to several configuration-oriented dialog boxes. The first dialog box focuses on importing settings from any previously installed version of Android Studio.

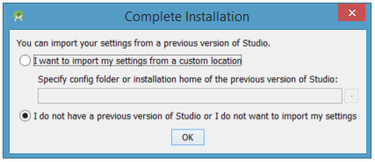


Fig.3.4.8. Import settings

If you're like me, and don't have a previously installed version, you can just keep the default setting and click OK. Android Studio will respond with a slightly enhanced version of the splash screen, followed by the Android Studio Setup Wizard dialog box:

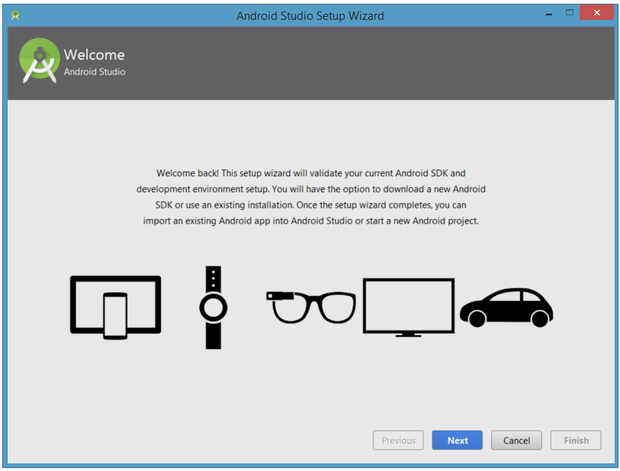


Fig.3.4.9. Validate your Android SDK and development environment setup

When you click next, the setup wizard invites you to select an installation type for your SDK components. For now I recommend you keep the default standard setting.

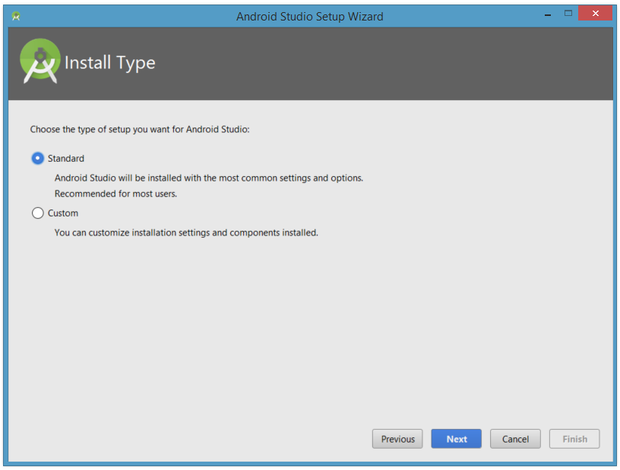


Fig.3.5.0. Choose an installation type

Click Next and verify your settings, then click Finish to continue.

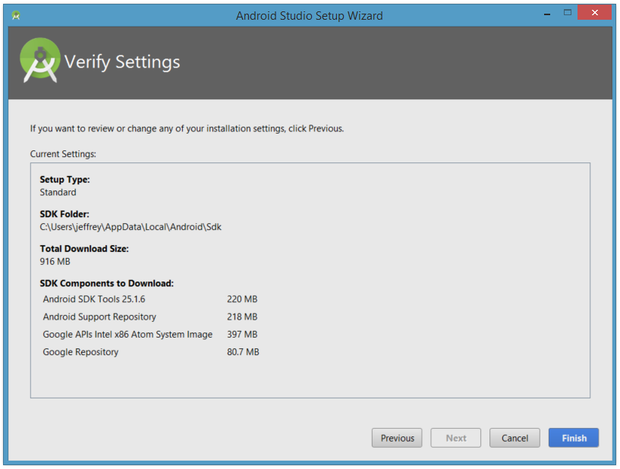


Fig.3.5.1. Review settings

The wizard will download and unzip various components. Click Show Details if you want to see more information about the archives being downloaded and their contents.

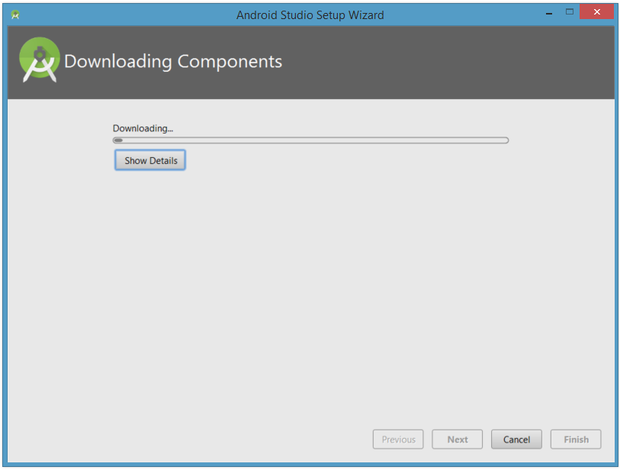


Fig.3.5.2. The wizard downloads and unzips Android Studio components

Finally, click Finish to complete the wizard. You should see the Welcome to Android Studio dialog box:

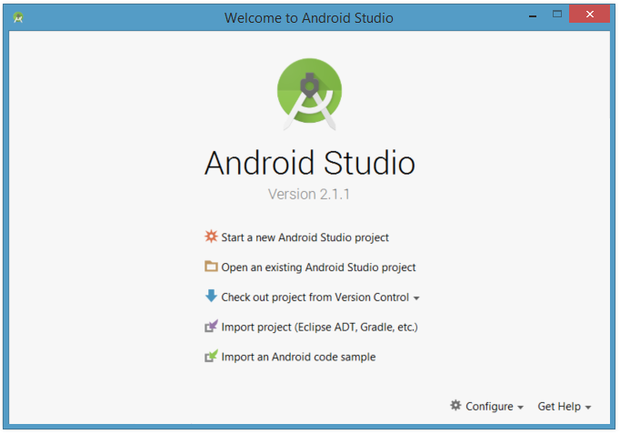


Fig.3.5.3. Welcome to Android Studio

You'll use this dialog to start up a new Android Studio project, work with an existing project, and more. You can access it anytime by double-clicking the Android Studio shortcut on your desktop.

## Starting a new project

From our setup so far, you should still have Android Studio running with the Welcome to Android Studio dialog box. From here, click Start a new Android Studio project. Android Studio will respond with the Create New Project dialog box as shown below.

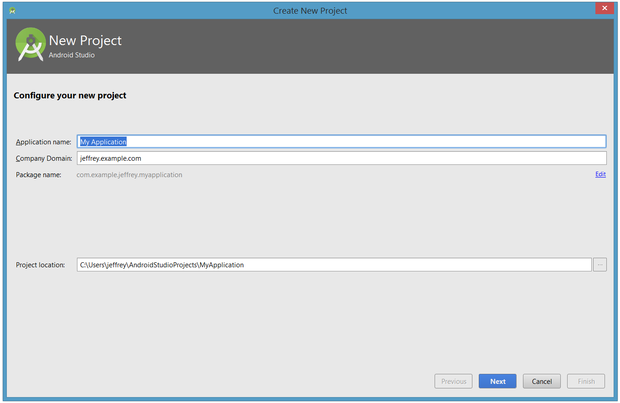
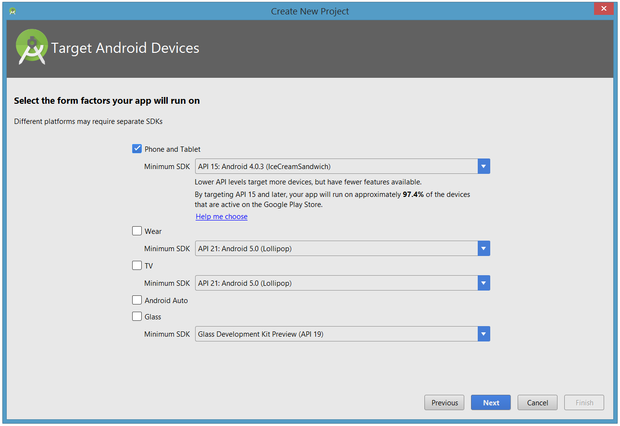


Fig.3.5.4. Create a new project

Enter W2A (Welcome to Android) as the application name and javajeff.ca as the company domain name. You should then see C:\Users\jeffrey\AndroidStudioProjects\W2A as the project location. Click Next to select your target devices.

Fig.3.5.5. Select your target device categories

Android Studio lets you select form factors, or categories of target devices, for every app you create. I would have preferred to keep the default API 15: Android 4.0.3 (IceCreamSandwich) minimum SDK setting (under Phone and Tablet), which is supported by my Amazon Kindle Fire HD tablet. Because Android Studio doesn't currently support this API level (even when you add the 4.0.3 system image via the SDK Manager), I changed this setting to API 14: Android 4.0 (IceCreamSandwich), which is also supported by my tablet.

Click Next, and you will be given the opportunity to choose a template for your app's main activity. For now we'll stick with Empty Activity. Select this template and click Next.

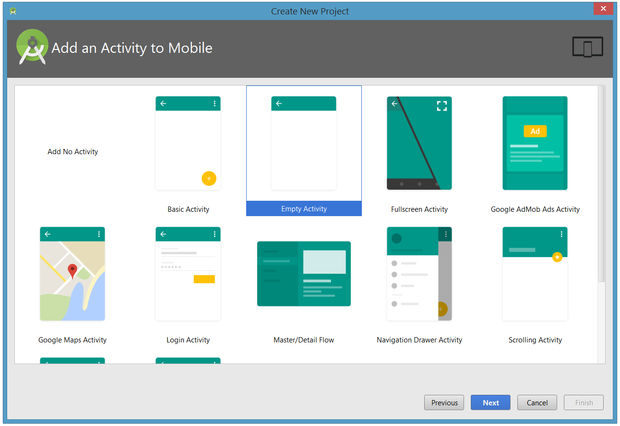


Fig.3.5.6. Specify an activity template

Next you'll customize the activity:

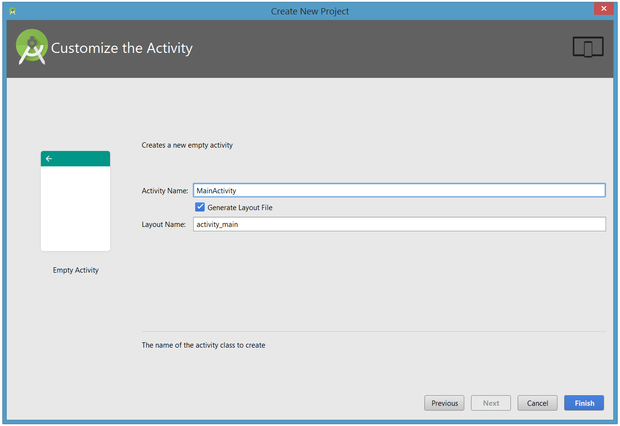
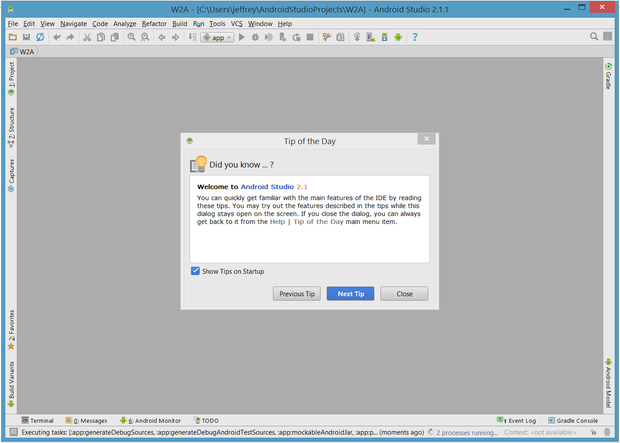


Fig.3.5.7. Customize your activity

Enter W2A as the activity name and main as the layout name, and click Finish to complete this step. Android Studio will respond that it is creating the project, and then take you to the project workspace.

Fig.3.5.8. Android Studio workspace

The project workspace is organized around a menu bar, a tool bar, a work area, additional components that lead to more windows (such as a Gradle Console window), and a status bar. Also note the Tip of the Day dialog box, which you can disable if you like.

## The project and editor windows

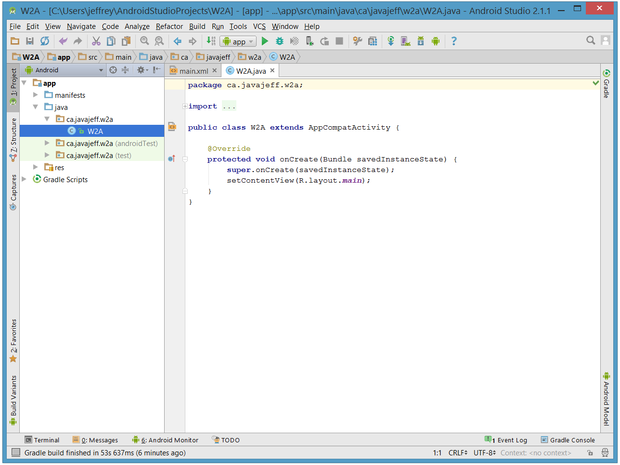
When you enter the project workspace, W2A is identified as the current project, but you won't immediately see the project details. 

Fig.3.5.9 The project and editor windows

**3.5 Conclusion:**

In this chapter we have given introduction about android os and had explained the software requirements specifications and in that user requirement, software and hardware requirements are mentioned. The installation procedure of android studio and algorithm and flow chart of the application process is mentioned.

**CHAPTER-4**

**DESIGN**

**4.1 Introduction:**

Every operating system or platform has guidelines for design. Consistent design helps to create a distinct look and feel for the operating system or platform. When you’re working on an application that will target Android, for example, you'll learn that there are a number of subtle and not so subtle differences regarding the vision that underlies the design. These are some examples of aspects which are emphasized differently than other mobile operating systems.

**Personalization:** The Android guidelines recommend that you include a level of personalization within an application as it helps make users feel at home. Giving the user the ability to theme an application is a good example of this concept.

**Icons over Words:** If you can communicate something through visuals, such as icons or images, then that should be the preferred method of communication. If you come across a scenario in which words are absolutely necessary, then make sure to keep it concise and actionable.

**Every User Is An Expert:** Mobile applications should always be easy to use. At the same time, you should give the user the impression that they're a power user. This can be accomplished by providing shortcuts or by implementing a powerful on boarding process.

**4.2 UML Diagram:**

UML stands for Unified Modelling Language. UML is a standardized general- purpose modelling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group.

The goal is for UML to become a common language for creating models of object oriented computer software. In its current form UML is comprised of two major components: a Meta model and a notation. In the future, some form of method or process may also be added to; or associated with, UML.

The Unified Modelling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modelling and other non-software systems.

The Unified Modelling Language represents a collection of best engineering practices that have proven successful in the modelling of large and complex systems.

The Unified Modelling Language is a very important part of developing objects oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

**GOALS:**

The Primary goals in the design of the UML are as follows:

1. Provide users a ready-to-use, expressive visual modelling Language so that they can

Develop and exchange meaningful models.

2. Provide extendibility and specialization mechanisms to extend the core concepts.

3. Be independent of particular programming languages and development process.

4. Provide a formal basis for understanding the modelling language.

5. Encourage the growth of OO tools market.

6. Support higher level development concepts such as collaborations, frameworks, patterns and components.

**Use Case Diagram:**

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use overview of the functionality provided by a system in terms of actors, their goals(represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.

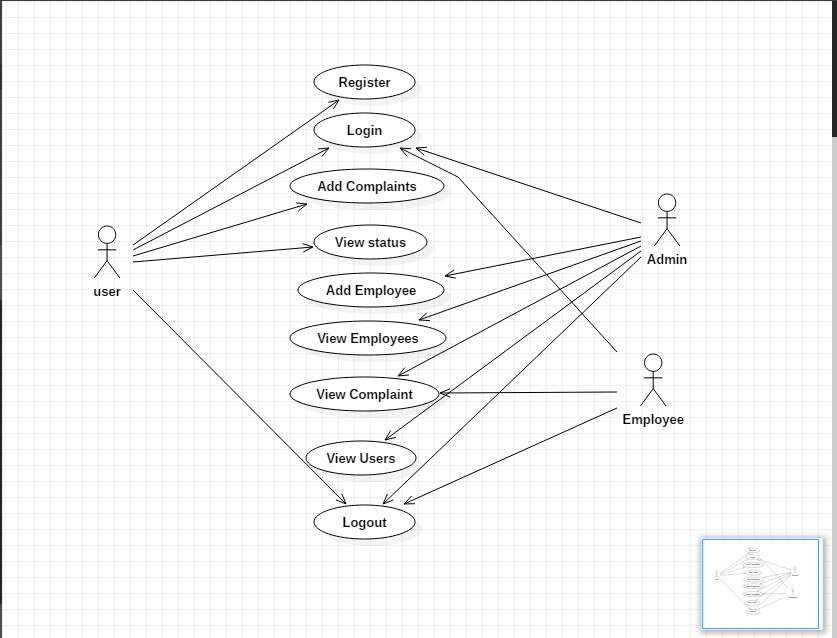


Fig 4.2.1: Use case Diagram

**Sequence Diagram**

A sequence diagram is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. A sequence diagram shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario. Sequence diagrams are typically associated with use case realizations in the Logical View of the system under development. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams.

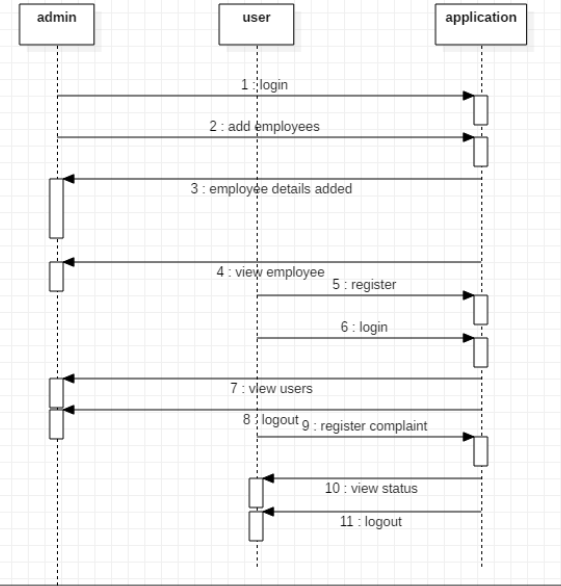
****

Fig 4.2.2: Sequence Diagram

**Class Diagram:**

The class diagram is the main building block of object oriented modeling. It is used both for general conceptual modeling of the systematic of the application, and for detailed modeling translating the models into programming code. Class diagrams can also be used for data modeling. The classes in a class diagram represent both the main objects, interactions in the application and the classes to be programmed. In the diagram, classes are represented with boxes which contain three parts:

* The upper part holds the name of the class.
* The middle part contains the attributes of the class.
* The bottom part gives the methods or operations the class can take or undertake.

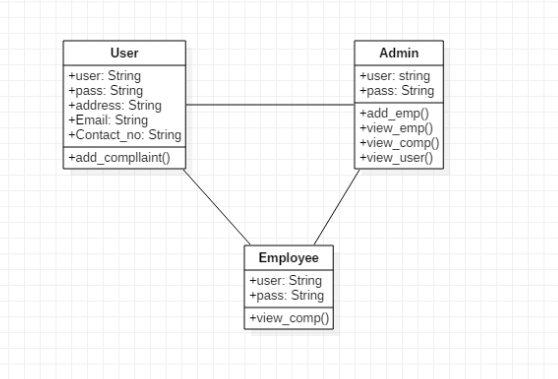
****

Figure 4.2.3: Class Diagram

**Activity Diagram**

Activity diagram is another important diagram in UML to describe dynamic aspects of the system. It is basically a flow chart to represent the flow form one activity to another activity. The activity can be described as an operation of the system.So the control flow is drawn from one operation to another. This flow can be sequential, branched or concurrent.

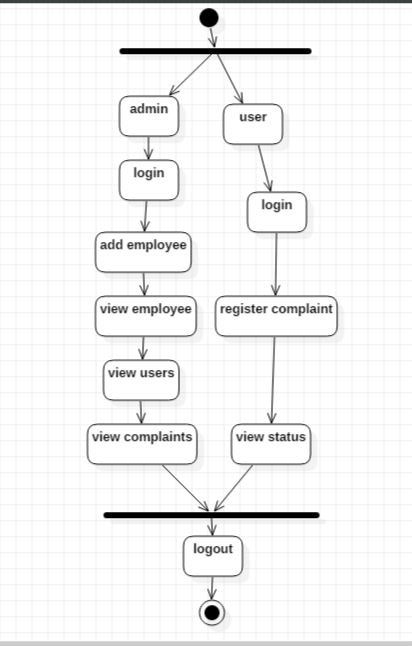


Figure 4.2.4: Activity Diagram

**4.3 Module Description**:

**User Module**

Citizens will have mobile application .The user has to create the account or sign in the account. If we have already created the account you can just sign in. While creating the account the user has to enter the username, mail id and password. After entering all credentials click on the create account button. She/he can register a complaint related to specific zone where he/she finds a problem. This application can provide a user friendly UI interface. Mobile application contains GPS tracking system too.

**Admin Module**

The admin has to create the account or sign in the account. Here the admin can acts as a user also. Admin can manage the accounts i.e, add the employees.

**Employee Module**

The employee has to sign in the account and view the complaints of the citizens and solve the complaints of the citizens where the problem is.

**Global Positioning System (GPS)**

Along with these request application will embed the location from which request are getting placed. This is going to work by GPS tracking system. GPS devices are wireless and use satellites to enable users to detect the position of mobile on earth.

**4.4 Conclusion**

Design of this application provides clear explanation of all modules and their functionalities with their UML diagram.AQ

**CHAPTER-5**

**IMPLEMENTATION & RESULTS**

**5.1 Introduction**

After designing the new system, the whole system is required to be converted into computer understanding language. Coding the new system into computer programming language does this.

It is an important stage where the defined procedures are transformed into control specifications by the help of a computer language. This is also called the programming phase in which the programmer converts the program specifications into computer instructions, which we refer as programs. The programs coordinate the data movements and control the entire process in a system.

It is generally felt that the programs must be modular in nature. This helps in fast development, maintenance and future change, if required.

The validity and proper functionality of all the modules of the developed application is assured during the process of implementation. Implementation is the process of assuring that the information system is operational and then allowing user to take over its operation for use and evaluation.Implementation is the stage in the project where the theoretical design is turned into a working system. The implementation phase constructs, installs and operated the new system.

**5.2 Explanation of Key functions:**

Code for account creation:

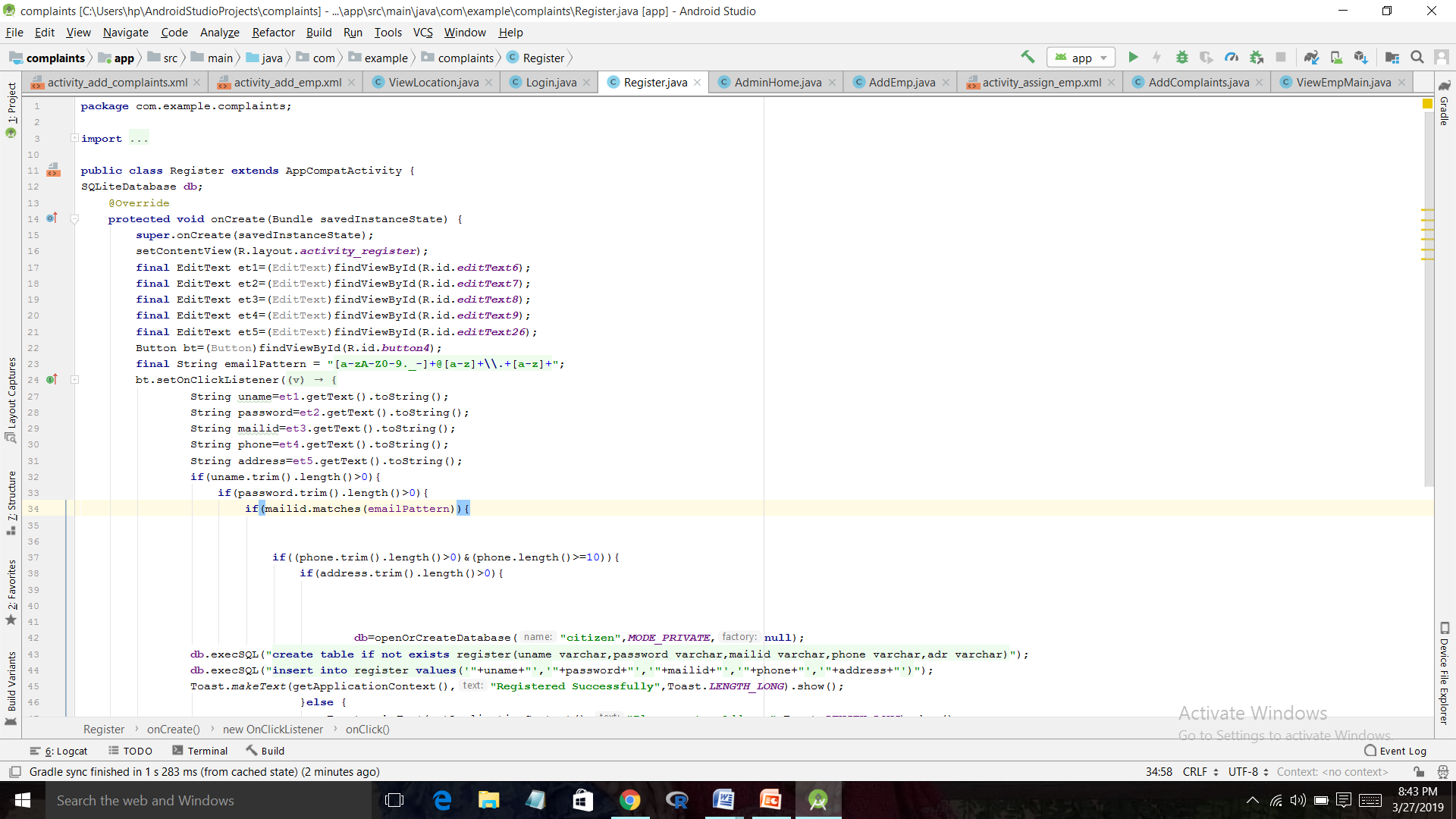


Figure 5.1 Code for account creation

**5.3 Method of implementation**

**5.3.1 Forms**

If the citizen wants to add the complaints or get status, he/she should login first. The user has to login with the username and password.

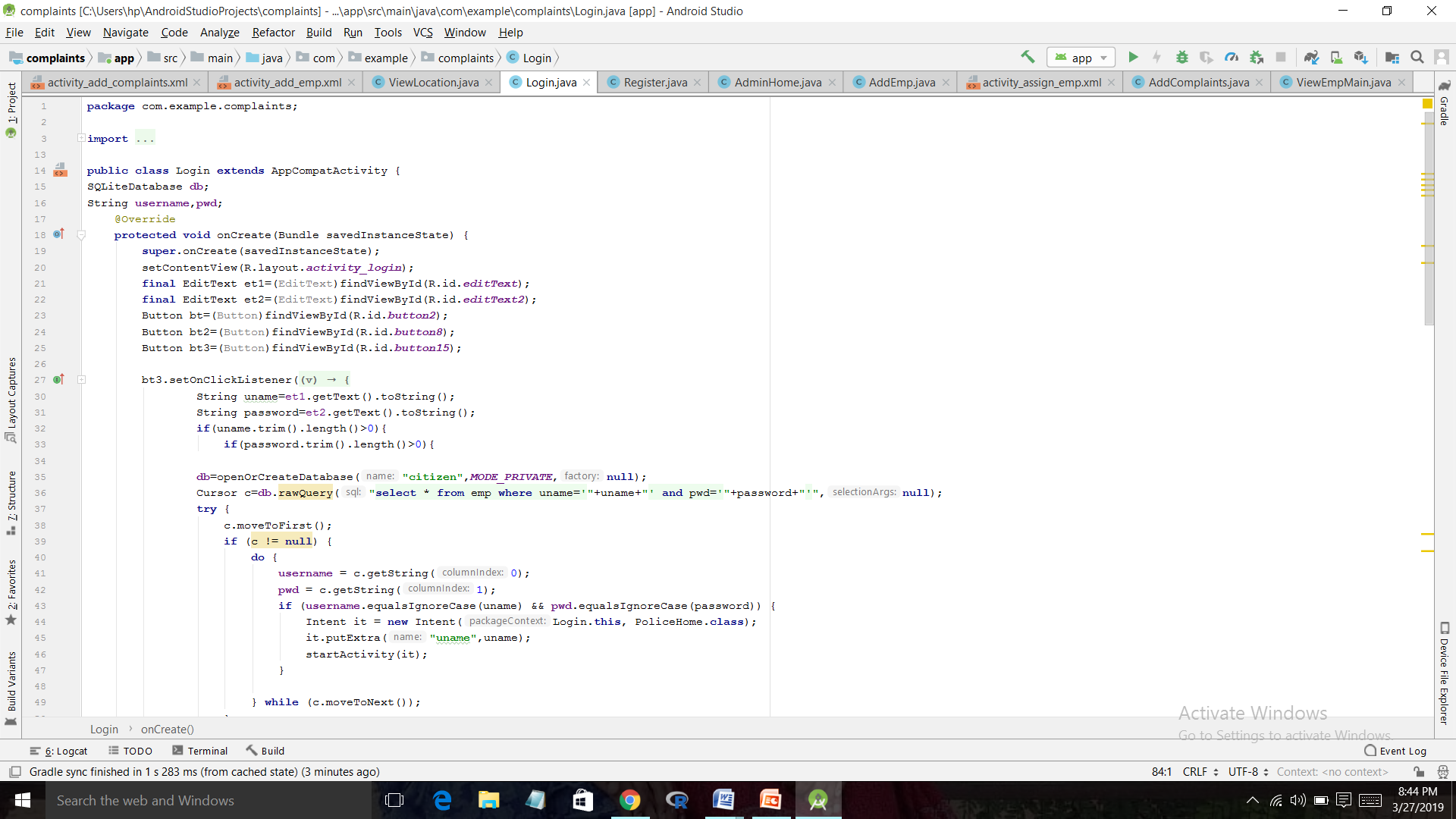


Figure 5.2: Code for login

After login the citizen has to add complaints

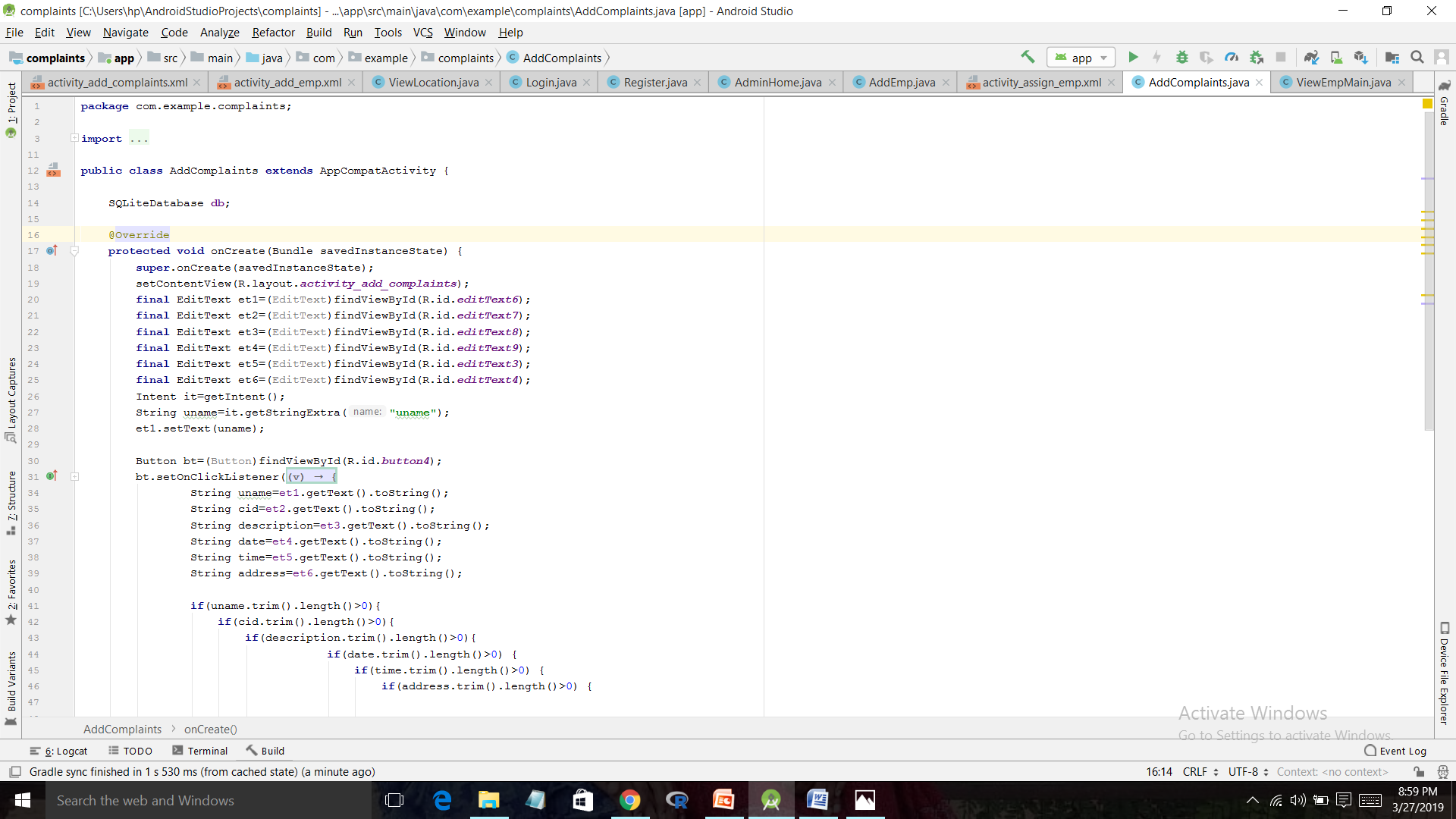


Figure 5.3: Code for add complaints

**5.3.2 Output screens**

Initially the user has to create the account, he/she has to enter the username, mail id and password.



Figure 5.4 Create account

If the user has already created the account, he/she can just login in with the account.

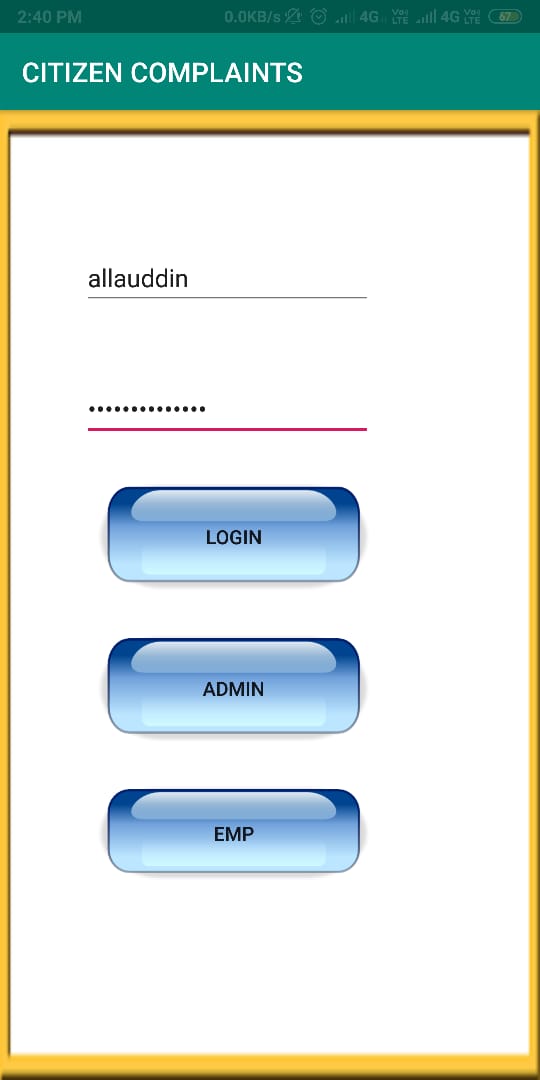


Figure 5.5: Login form

After login in the account the user has to add the complaints where the problem is



Figure 5.5: Add Complaints

**5.4 Conclusion**

In this chapter we have described about the implementation and results, the explanation of key functions have been specified.

**CHAPTER-6**

**TESTING AND VALIDATION**

**6.1 Introduction:**

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub-assemblies, assemblies and/or a finished product. It is the process of exercising software with the intent of ensuring that the software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

**6.2 Design of Test cases and Scenarios:**

**6.2.1 Types of Tests:**

**6.2.1.1 Unit testing:**

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application. It is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive.

Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration.

**6.2.1.2 Integration testing:**

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

**Functional test:**

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input: identified classes of valid input must be accepted.

Invalid Input: identified classes of invalid input must be rejected.

Functions: identified functions must be exercised.

Output: identified classes of application outputs must be exercised.

Systems/Procedures: interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, and special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

**6.2.1.3 System Testing:**

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

**White Box Testing:**

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

**Black Box Testing:**

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box .you cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works.

**Unit Testing:**

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

**Test strategy and approach**

Field testing will be performed manually and functional tests will be written in detail.

**Test objectives**

* All field entries must work properly.
* Pages must be activated from the identified link.
* The entry screen, messages and responses must not be delayed.

**Features to be tested**

* Verify that the entries are of the correct format
* No duplicate entries should be allowed
* All links should take the user to the correct page.

**Integration Testing**

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects. The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at the company level – interact without error.

**Acceptance Testing:**

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

**6.3 Test cases**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **SLNO** | **SCENARIOS** | **EXPECTED RESULTS** | **ACTUAL RESULTS** | **STATUS** |
| 1 | InstallDroidLocator.apk  File on Android phone | Installation  Successful | Installation Successful | Success |
| 2 | Check whether UI is Displaying On screen | Display UI | Display UI | Success |
| 3 | Check whether username and passwords are stored. | Stored Successfully | Stored successfully | Success |
| 4 | Check whether employees are assigned to users. | Assigned successfully | Assigned successfully | Success |
| 5 | Check complaints are added by users | Added successfully | Added successfully | Success |
| 6 | User check status | Checked successfully | Checked successfully | Success |

**6.4 CONCLUSION**

Testing and validation gives the correctness of the output of the application by using some testing techniques.

**CONCLUSION**

This project provides a direct communication between the citizen and the municipal corporation. This will again help in registering the problems that one is facing in particular area and by continuously following up them will result in a good, clean and peaceful environment.

**REFERENCES**

**Journal**

[1] Trupti Bomble and Ritika Raut, “Android Based Complaint Management System for Municipal Corporation”, Int. Journal of Engineering Research and Applications., Volume-5,Issue 4,pp.64-66,April 2015.

[2] Gausar Gawde and Sharvil Sartare, “Mobile Application for Resolving Citizen Complaints”, ISARCET., Volume 5, Issue 3, March-2016.

**World Wide Web**

[3] Rich Miner, Nick Sears ,”Android Developer”, October 2003

<https://developers.google.com/vision/android/getting-started>

[4] https://developer.android.com/studio