ТЕХНИЧЕСКИ УНИВЕРСИТЕТ – ВАРНА

ФАКУЛТЕТ ПО ИЗЧИСЛИТЕЛНА ТЕХНИКА И АВТОМАТИЗАЦИЯ

Катедра „Софтуерно Инженерство “

A blue and white logo

Description automatically generated with medium confidence

**Курсова Работа по**

**АБТМУ**

Android базирани технологии за мобилни устройства

**Тема:**

Приложение за съобщения.

**Разработил:**  **Проверил:**

Ивайло Пламенов Руменов  ас.Велислав Колесниченко

Фак. № 23651227

# Задание

Заданието гласи да се проектира и програмира приложение за комуникация състояща се от съобщения. Което да включва потребителски профили помежду които ще се състои комуникацията.

Функционалности на приложението трябва да включва създаване на потребителки профил, добавяне на приятели, съобщения помежду приятели и групи от потребители.

Приложението трябва да се свързва със отделен сървър който приема и обработва входните съобщения. Всички съобщения и информация за потребителя да се съхраняват в релационна база от данни PosgreSQL.

Трябва да се разгледа и документира подход за имплементация на приложението. Да се разработи работещо демо и потребителска документация към него.

# Увод

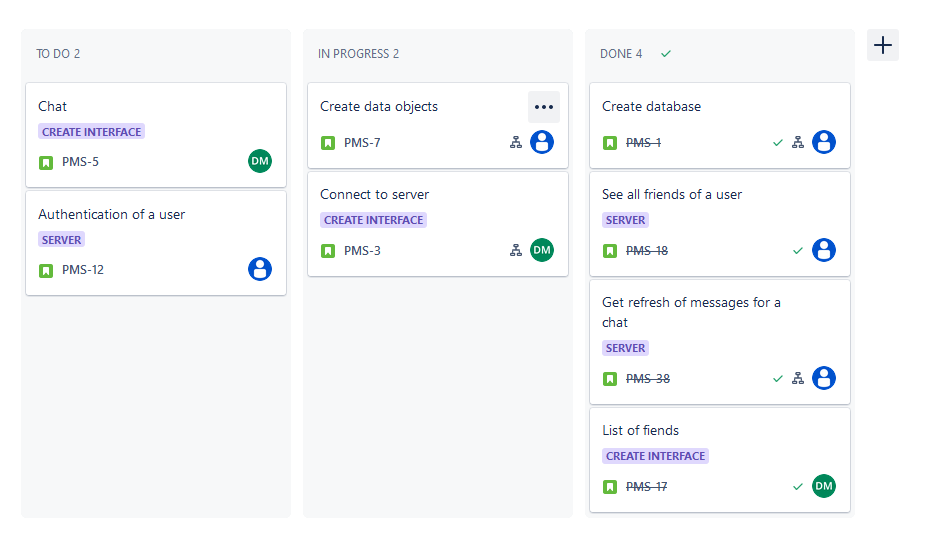
Разгледано в курсовата работа е комуникационно приложение за изпращане на съобщения помежду потребители. Приложението е разработено на Android SDK версия 34 и подържа инсталация до Android SDK 31. Използвано е Android Jetpack Compose методология за разработване на приложение. Разработването на проекта е реализирано в три главни части: планиране, реализация, документация.

# Планиране

Планиране на проекта е ключова част при започването на ново задание. Тук се разглежда всичко, от функционалности, тяхната сложност до времето нужно за реализацията им.

Планирането на проекта е реализирано с помощта на уеб базираното приложението Jira, което е световно известно с неговото умение да подобрява процеса за проектирани на програмни продукти. Методологията за реализиране на продукта разчита на Agile. Agile представлява гъвкава методология за реализация на програмни продукти. Тази методология работи на база спринтове, които обслужват една главна епопея коя представлява цялостния проект или поне част от него. Затова всички функционалности, които трябва да се имплементира се записват като истории към някоя епопея.

Затова е са разписани всичко малки задачи нужни за разработване на една функционалност. Така може лесно и бързо да се оцени за колко време ще е нужно за разработването на всяка функционалност и от там на цялостния проект.

За всяка една история, която представлява индивидуална функционалност, е оценена помежду 8 и 12 работни единици. Като е прието че една работна единица е оценена на два работни часа. Направени са седем на брой функционалности за реализирани. Те включват: регистриране на потребител, вписване на потребител, добавяне на приятел, визуализиране на всички приятели, визуализиране на всеки чат групи в които потребителя членува, изпращане на съобщение, визуализиране на съобщения от приятел или група. Общия брой работни единици е оценен на осемнадесет и пет което преведено в работни часове са сто шейсет и осем, и това преведено в работни дни по осем часа на работен ден е естимирано за реализацията на проекта двадесет и един работни дни.

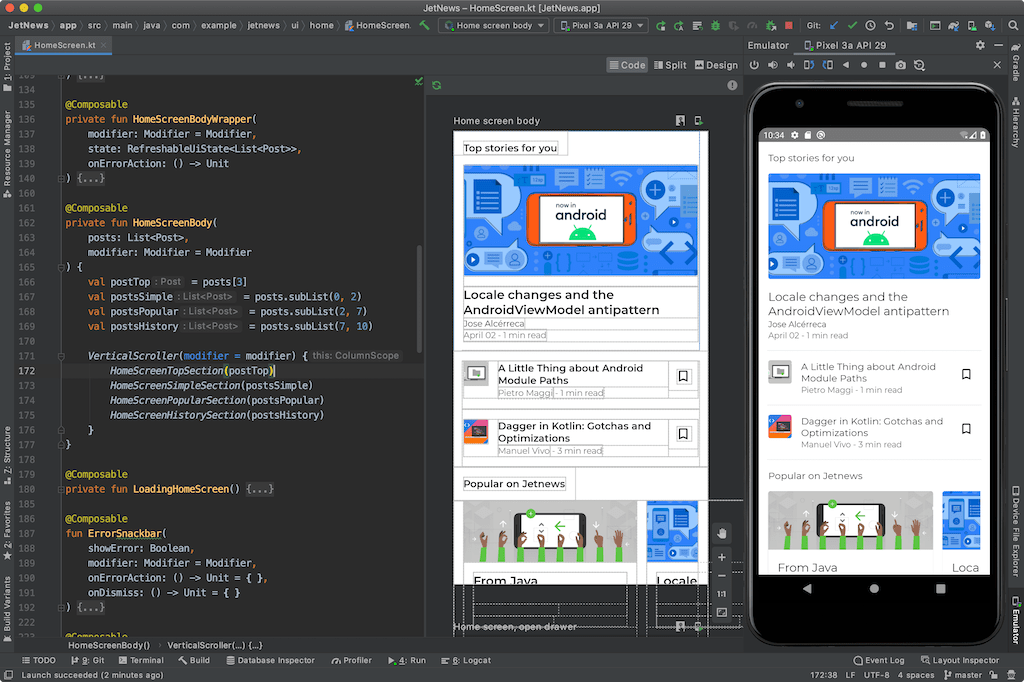
Фиг. № 1: Изображение на проекта в неговите начални стадий.

# Конкурентно проучване

Относно конкурентното проучване първо трябва да се оцени в коя маркетинг среда ще е приложението. Знаем че положението ще служи само и единствено за комуникация помежду потребители. Тоест можем да изключим че е от рода на социалните мрежи. То в неговото описание е по утилитарно. Затова можем да го постави в семейството на специализираните приложения за съобщения. Което за пазара може едновременно да е добре и зле. В това семейство на специализирани приложения за чат влизат: Diskor, Messenger, Apple Messages, TeamSpeack. Това семейство от специализирани приложения крият функционалности търсени от много потребители на подобни приложения което може да провали изцяло пазарния успех на приложените.

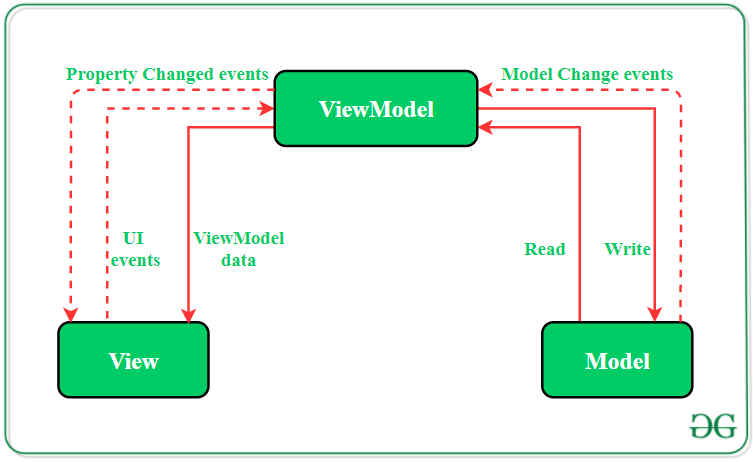
# Проектиране

Проектирането е частта където след анализ на проблемите се асоциира и се приема, и при нужда се пременя планът, това е заложено в Agiel методологията. По време на проектирането е и прието широко обучение при липса на знания свързани с андроид и какви конкретни методи да се ползват за реализиране на конкретна задача.

При започването на проекта е създаден Android jetpack compose приложение на SKD 34. Android jetpack compose e новият метод на писане на потребителския интерфейс на андроид. Той е тотално променен от последните методи на използващи xml за определяне на как ще изглежда потребителския интерфейс. Заменено е с директивни Kotlin functions които изрисуват елементите на екрана. Както и новата методология се поддържа само и единствено от езика Kotlin който е определен за предпочитане от Google и всякаква нова функционалност разработена от Google ще се подържа от Kotlin. Java дълги години доминира като главен език в андроид пространството но в днешни дни отпада бързо заради обновленията от google.

Фиг. № 2: Пример за визуализиране чрез jetpack compose.

Започвайки да се програмира приложението е решено да се използва само едно Activity и към него множествени Views( изгледи). Така информацията на екрана е консистентна помежду различните изгледи. Този начин на персистиране на информацията помежду различни изгледи се нарива View-Model-ViewModel. Където модела(Model) са обектите от данни, които съдържат специализирана информация за даден бизнес обект. View представлява потребителския интерфейс или така наречения изглед. ViewModel е обект променящ и персистиращ моделната информация за изгледа. Тоест когато потребителя прави промени по интерфейса то тези промени се обработва от viewmodel които е отговорен за тях и се променя спрямо желанията на клиента.



Фиг. № 3: Графика на методологията View Model ViewModel.

Достигането на данни от сървъра се случва чрез repository. Този клас служи само и единство за приема на данни от сървъра. Поради метода TCP който е избран за комуникиране със сървъра е то се случва по синхронен начин, което означава че всяка заявка се отваря нова нишка където ще се изпълни дадената заявка и се очаква отговор в viewmodel. След което viewmodel изчака информацията да пристигне от заявка към сървъра то информацията на потребителския изглед ще се промени. Това позволява на множество премени да се състоят едновременно и потребителя да продължава да ползва приложението докато чака дадената задача която желае. За реализиране на списъка с приятели и сторено точно това. Една синхронна заявка е използвана за да зареди информацията от сървъра.

# Бъдещи подобрения

Когато стане дума за подобрения винаги има какво да се желае. Това се отнася и за програмните продукти. Един програмен продукт в днешни време се проектира бързо и се надгражда по време на неговото експлоатация. Това се държи на методологията Agile която се практикува често в програмните компании. Както и бизнес тактиката за First to market. Която гласи че: няма значение колко е недовършен продукта, той трябва да бъде в публичното око. Което е приложимо и към големи корпорации както и за малки start up бизнеси.

За в момента е осъзнато че липсват функционалности, които повечето чат приложения на пазара имат и са нужни за официално чат приложение.