GitHub Username: MohNage7

GameMe

Description

GameMe is community for gamers. Where they can communicate with each other and ask any question comes in their mind.

Intended User

This app is for gamers.

Features

List the main features of your app. For example:

- Registration
- Login
- Create post
- Like post
- Upvote / downvote questions
- Upvote / downvote answers
- Like replays
- Comment on a post
- Like comments / replays
- Mark post as answer

Common Project Requirements

- App will be written solely in the Java Programming Language
- App will utilize stable release versions of all libraries, see below table.

Name	Version
Android Studio	3.1.3
Gradle	3.1.3
ButterKnife	8.8.1
FirebaseUI Realtime Database	4.1.0
Firebase Realtime Database	16.0.1
Firebase Authentication	16.0.2
Glide	4.7.1
CircleImageView	2.2.0
Design support	27.1.1
Firebase Storage	16.0.1

Material Design

- App theme will extends AppCompat
- App will use an app bar and associated toolbars.
- App will use standard and simple transitions between activities.

Core Platform Development

- App will keep all strings in a `strings.xml` file and enables RTL layout switching on all layouts.
- App will validate all input from servers and users. If data does not exist or is in the wrong format, the app logs this fact and does not crash.
- App will include support for accessibility. That includes content descriptions, navigation using a D-pad, and, if applicable, non-audio versions of audio cues.

User Interface Mocks

Registration screen

If there user doesn't have an account. He can create one from this screen by filling the required fields with valid informations and then press Register now.

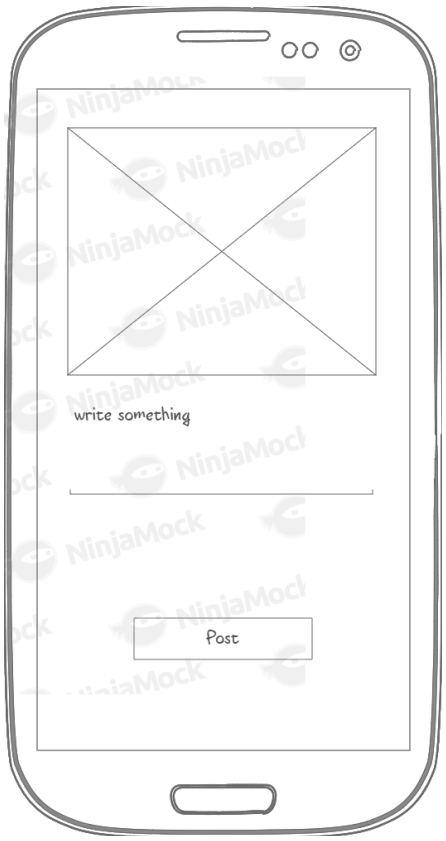
Username screen

After successful registration user should create his username. it is required and it should be unique for each user.

Login screen

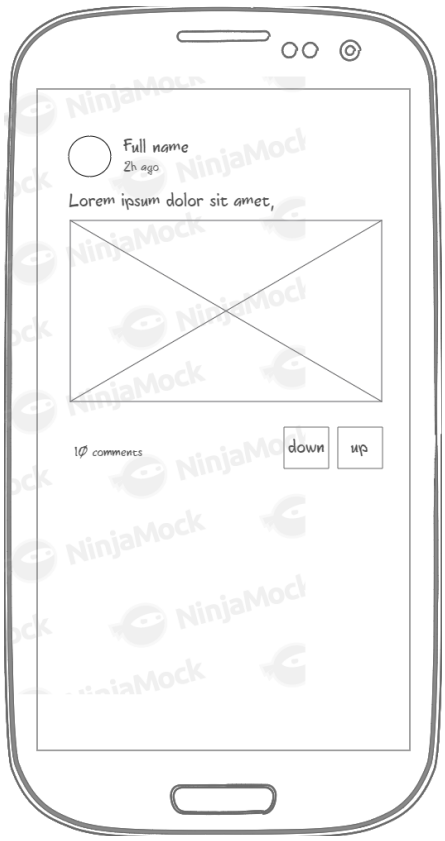
User can login with his email and password. If he already created an account.

New post screen



Users can create new post. Posts can has three types (text , text and image , image). If the post is image only. Answer feature will be disabled.

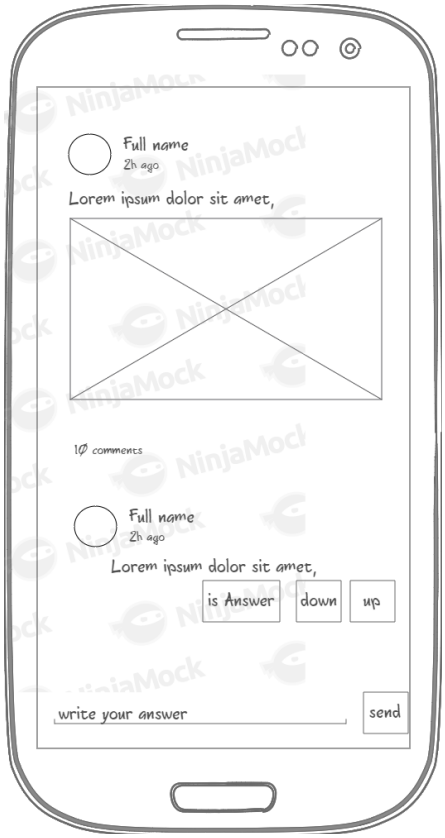
Home screen



Users can interact with other user's posts. Though Home screen or details screen. With the following action in home screen.

- Up vote
- Down vote
- View comments
- Navigate to user's profile

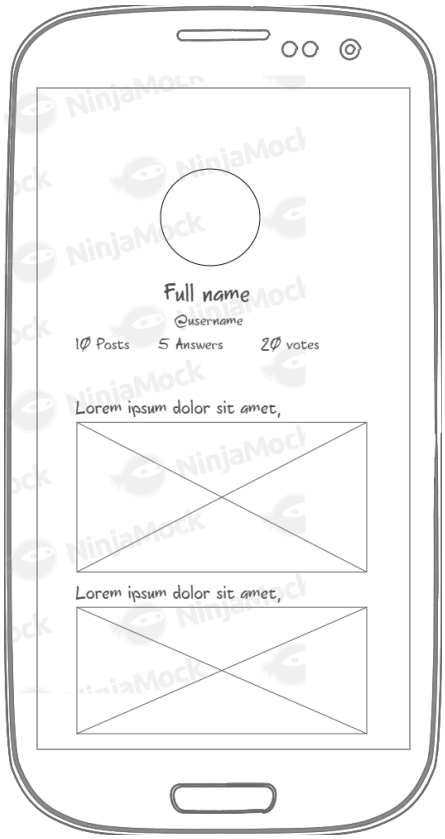
Post details screen



Users can interact with other user's posts. Though Home screen or details screen. With the following action in details screen.

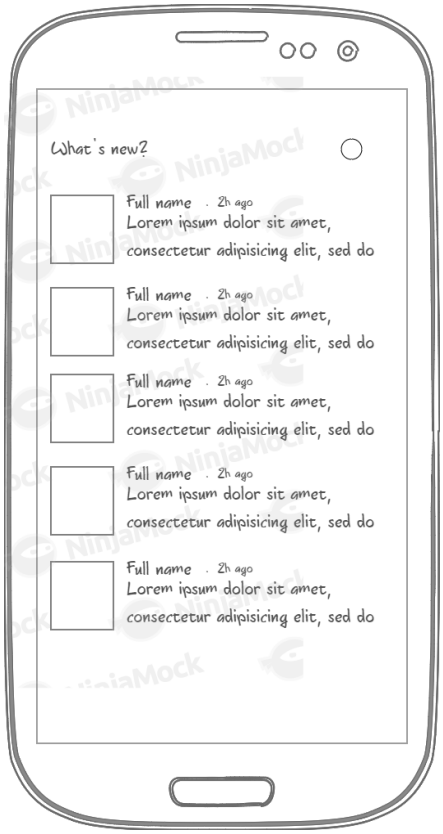
- View all comments
- Send new comment
- Up vote
- Down vote
- Mark as answer
- View comments
- Navigate to user's profile

Profile screen



Profile will contain user's basic information with simple statistics about his interaction with the community (posts count , answers count and recieved votes count). And of course user's posts.

Widget screen



Widget screen will display latest 5 posts and button to navigate to profile screen.

Key Considerations

How will your app handle data persistence?

Users posts and personal data will be saved in Firebase Realtime database.

Describe any edge or corner cases in the UX.

Project's UX is straight forward. There's no corner cases to describe.

Describe any libraries you'll be using and share your reasoning for including them.

- Glide: handle the loading and caching of images.
- Firebase Authentication : enable registration and login in the app
- Firebase Realtime Database: store and sync user's data in real time
- Firebase Storage : store media
- Firebase UI for Realtime Database: load and sync posts
- Scissors : Image cropping library to enable users to edit their images before uploading them
- ButterKnife: For fields and method binding for android views.

Describe how you will implement Google Play Services or other external services.

The project will be using the following 3 services from Firebase.

- Firebase Authentication : enable registration and login in the app
- Firebase Realtime Database: store and sync user's data in real time
- Firebase Storage : store media

Integration steps

- Create new firebase project
- Download google-services.json file and put it in app's directory to connect the app with firebase project
- Configure gradle to be able to use the services.

Next Steps: Required Tasks

Task 1: Project Setup

- Create new Android project.
- Configure project with required libraries and services.
- Create new Firebase project.
- Link firebase with the android app.
- Configure Release build type.

Task 2: Implement UI for Each Activity and Fragment

- Build UI for Splash screen.
- Build UI for Registration screen.
- Build UI for Login screen.
- Build UI for username screen.
- Build UI for New post screen.
- Build UI for Home screen.
- Build UI for Profile screen.
- Build UI for Post details screen.
- Build UI for widget.

Task 3: Implement splash Activity logic

Splash screen will be for checking user's registration status.

- Init views and onClickListeners with butterKnife
- Check if the user is logged in before
- Navigate user to Login screen or Home screen

Task 4: Login Activity logic

Login activity will be parent view for (Registration, sign in and username) fragments

- Init views and onClickListeners with butterKnife
- Set default fragment
- Handle replacing the fragments
- Handle back press

Task 5: Sign in Fragment logic

Sign In fragment will be default fragment for Login activity.

- Init views and onClickListeners with butterKnife
- Init Firebase Database and Auth
- Validate user's data locally
- Send user's data to server for Auth
- Notify user with success or fail results
- Navigate to Home Activity on success

Task 6: Registration Fragment logic

- Init views and onClickListeners with butterKnife
- Init Firebase Database and Auth.
- Validate user's data locally.
- Send user's data to server to create new user.
- Notify user with success or fail results.
- Navigate to username Fragment on success.

Task 7: Username Fragment logic

- Init views and onClickListeners with butterKnife
- Init Firebase Database and Auth
- Validate username
- Validate username against other users to prevent duplicate
- Notify user with success or fail results
- Edit user's info in server with the new username on success
- Navigate to Home Activity on success

Task 8: Home Activity logic

Home activity will be parent view for (Home , Profile) fragments with bottombar that enable access to HomeFragment , ProfileFragment and NewPostActivity.

- Init views and onClickListeners with butterKnife
- Init Bottom bar and init it's listener
- Set default fragment
- Handle replacing the fragments
- Handle back press

Task 9: Home Fragment logic

Home fragment will be default fragment for Home activity.

- Init views and onClickListeners with butterKnife
- Init Firebase Database
- Init FirebaseUI Database
- Init Bottom bar and init it's listener
- Create Posts ViewHolder and init views and actions in it.
    1. Handle up vote and down vote. if it's a question.
    2. Handle like if it's normal post
    3. Handle navigate to user's profile if clicked
    4. Open Details screen if post is clicked

- Load users posts using FirebaseUI

Task 10: Profile Fragment logic

- Init views and onClickListeners with butterKnife
- Init Firebase Database
- Display user's informations
- Create User's Posts ViewHolder and init views and actions in it.
    1. Fill views
    2. Open Details screen if post is clicked

- Load user's posts using FirebaseUI

Task 10: Posts Details Activity logic

- Init views and onClickListeners with butterKnife
- Init Firebase Database
- Display posts information
- Handle up vote and down vote. if it's a question.
- Handle like if it's normal post.
- Navigate to users profile if username, user real name or user's profile picture
- Enable users to send new Comment / Answer
- Create Comments / Answers ViewHolder.
    1. Enable post owner to mark question as answer if it's a question.
    2. Enable users to like comments if it's normal post
    3. Enable users to upvote or downvote answer if it's question
    4. Enable replying to users comments / Answers
    5. Enable tagging users

- Load users comments / answers using FirebaseUI

Task 10: Create Widget

- Implement Widget logic and data sync.
- Widget screen will display latest 5 posts and button to navigate to profile screen.
- App will use IntetService to fetch data and show it on Widget.

Task 11: Test and build release.

- All app dependencies will be managed by Gradle.
- App will be equipped with a signing configuration, and the keystore and passwords are included in the repository. Keystore is referred to by a relative path.
- App builds and deploys using the installRelease Gradle task
- App builds from a clean repository checkout with no additional configuration.
- Generate releaseand debug APK version.
- Test the app before launching.