SLIDING PUZZLE APP (ANDROID)

REQUIREMENTS AND PLANNING DOCUMENT

J.E.K.A. SOLUTIONS

KERSLEY OSEMEKHIAN JATTO

JESSA LASPESA

ARTURO MARTINEZ

ELLEN STUDER

Requirements Specification

1. Purpose and Scope

The purpose of this project is to create an Android application for our client – Professor Franceschi. Our client would like an application that is similar to the sliding puzzle game where the game-board is broken up into different tiles. The user will then have to solve the puzzle by swapping two pieces. Once the puzzle is back to the original state of the picture, the user has won the game.

1.1 Stakeholders

- **The client** that wishes to have the Android app completed.
- **Users/players** that would be downloading and playing the app.
- **Developers** who wish to complete a quality app and fulfill the contract with the client. We will also like to complete code that is easy to understand that can be reused in the future; in addition, code that can be easily maintained in the future for further development.
- **Potential investors** who will look to invest money into the app, as long as it is a quality product and can return a profit.

1.2 What is within scope and what is out of scope?

In Scope

- User/player will be able to choose whether to play or create a game
- The app needs to be done as an Android application
- Choosing a grid size for the puzzle
    - The options are between 3x3, 4x4, or 5x5
- Puzzle will need to be scrambled before starting game
- User/player will be able to choose what picture they want to play with
    - Picture can come from their phone's camera roll
    - Picture can come from taking a picture from the phone's camera
    - Picture can come from a database
- Game-board will be able to detect when a tile is touched
    - A tile swap will occur when a tile is held and dragged over another tile – once let go, those two tiles will switch spots

- The app will need to notify the player when they have successfully completed the puzzle
- When puzzle is completed, should be taken back to the home screen
- Pictures will show the fastest time that particular puzzle was completed in and by whom
    - Players will be able to enter their name if their time is the new best time
- A name is attached to the picture of who submitted it to the database

Out of Scope

- Have ads incorporated into the app
- A logo for the app icon

2. The problem and the problem domain

Problem

At startup of the app, the user will have the option of playing the game or creating a puzzle. If the user chooses to play the game, then they will have the option of choosing a picture from a database, choose a photo form their camera roll, or using their smartphone to take a picture that will be used to play. If the user elects to create a puzzle, then they will use their smartphone to snap a picture, which will then be sent to the database; they will also be able to attach their name so that other players will know who submitted the picture. From there, other users will be able to go through the databases of pictures submitted by users to use as their puzzle. The game-board will be of three different sizes and will be chosen by the user/player before the game starts. They will have the choice of a 3x3, 4x4, or 5x5 grid. The picture that was chosen will be split into the corresponding grid and the tiles will be scrambled. To solve the puzzle, the user will have to click and drag tiles. In order to swap a tile, the original tile that is being dragged will need to be placed over the tile where the user wants to swap their tile to. Once the puzzle has been successfully complete, the app will let the player know. If the player completes the puzzle with the best time for that picture, then they can attach their name along with the new best time for that picture. Otherwise, they will be taken back to the home screen.

2.1 Problem Domain

The problem domain for this project will be developing the code to function as an Android application. Furthermore, the development of the application will need to be done in Android studio; where, there could be some growing pains as the developers are still learning the different functions which Android studio is capable of.

2.2 App Gameplay Flowchart

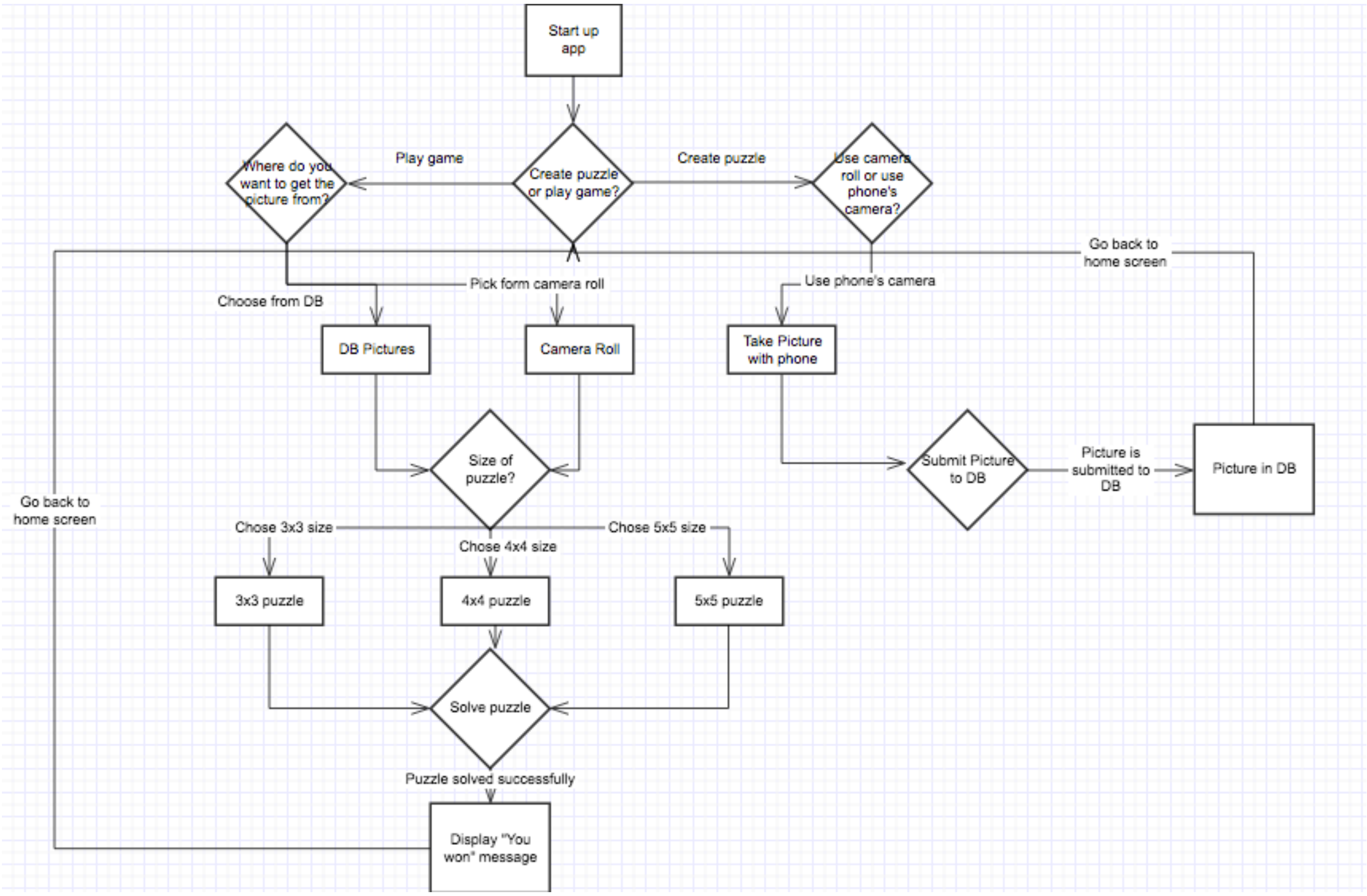

figure 1.1

3. Glossary

- **Android Studio**: Android based development kit for creating mobile apps that uses Java and XML
- **Puzzle**: A game that will be made by splitting a picture into a 3x3, 4x4, or 5x5 board. The board will be made with a user's photo or from a picture saved in the database
- **Loyola Server**: Loyola's cloud through which user photos will be stored and transferred
- **User Account**: A hub the user uses to access the game and allows a user to access their puzzles or the photos in the phone. Will also record a user's game statistics like quickest puzzle completion
- **User**: a player of the game
- **Smartphone**: A communication device that can take photos and will be the medium the game is played on
- **Emulator**: Android Studio's electronic model of an android phone that will be used to test the game during programming
- **Picture**: A graphic that the user either takes themselves or chooses from the game server
- **Grid Size**: The number of pieces that a photo will be split into. The number of pieces on each side will be equal regardless of the chosen size
- **Difficulty**: How challenging the puzzle will be. The difficulty increases with the number of pieces

4. Functional Requirements

4.1 The primary actors and their general goals

Primary Actor

- **User/Players**
  - Creates new puzzle
  - Solves puzzle
  - Complete puzzle, submit name if best time

Functional Requirements

- A new user needs to be able to create an account.
- An existing user needs to be able to retrieve a forgotten password.
- An existing user needs to be able to sign in.
- An existing user needs to be able to upload a picture to the database.
- An existing user needs to be able to choose a picture from the database.
- An existing user needs to be able to complete the game from a chosen picture
- An existing user needs to be able to refresh a game if they get stuck
- An existing user needs to be able to access their game statistics.
- An existing user needs to be able to access instructions.
- A player can select difficulty of the puzzle
- A player can slide a piece to swap it with another piece.

4.2 List of features of your software system

High Priority

- Upon opening the app for the first time, a user shall be prompted to make an account or sign in.
- The home screen shall prompt the user to "choose a photo" or "upload a photo"
    - If the user wants to upload a photo, they shall be able to use their camera or select from their camera roll
- If the user wants to choose a photo, they shall be taken to the database of pictures
- A help page with instructions shall be available at any point in the app **besides** the choosing or uploading of a photo
- During the game, the user shall slide a tile to swap it with another tile.
    - When the user moves a tile more than half way over another, it shall snap to the swapped location.
- When the puzzle is solved, the user shall be alerted and asked if they would like to retry the same puzzle or try a new puzzle.
- If the user is stuck on the puzzle, they shall have the option to shuffle the puzzle and start over.

- The user shall be able to return to the home screen from any internal page.

Lower Priority (wish list)

- The user shall be able to “retrieve forgotten password” on the sign in screen
- The user shall be able to select different difficulty levels after choosing a picture
- The user shall be able to access their saved statistics from the home screen
  - The saved statistics will include best time for each difficulty level
  - The saved statistics will also include least number of moves for each difficulty level
  - The saved statistics will also include the number of puzzles completed/uploaded
- The user shall be able to re-shuffle the game by shaking the phone
- The game will have sound effects for better accessibility

4.3 Use Cases

Use Case ID:	001		
Use Case Name:	Primary User-Puzzle Creator		
Created By:	Jessa & Ellen	Last Updated By:	Jessa & Ellen
Date Created:	10/19/2016	Date Last Updated:	10/19/2016

Actor:	**Primary User / Puzzle creator**
Description:	Primary User’s submits a picture to the database for use as a puzzle
Pre-conditions:	User has account and can log in to the app. User has photo in library • *Or* - User has ability to take a picture with their device User has internet connectivity.
Post conditions:	The picture appears in the database/global selection screen for other user’s to see and use
Priority:	High

Normal Course of Events:	1. Primary User’s is given the option to take a picture or select one from the camera roll 3. If a user selects to take a picture, the camera opens 4. If a user selects to choose a picture, the camera roll opens 5. User submits the picture (either the one they took or the one they selected) 6. Picture appears in the global selection view with the user’s name underneath
Unsuccessful vetting result	· The user attempts to upload a picture without internet connection · A error screen pops up and asks user to connect to the internet · Picture is not added to the global selection screen · User is redirected to step 1
Notes and issues:	If the user is a first-time user, they must give permission for app to access camera and camera roll. User must have an account.

Use Case ID:	002		
Use Case Name:	Primary User-Puzzle Player		
Created By:	Jessa & Ellen	Last Updated By:	Jessa & Ellen
Date Created:	10/19/2016	Date Last Updated:	10/19/2016
Actor:	**Primary User**		
Description:	User chooses a photo from the database and plays the puzzle game.		
Pre-conditions:	User has an account and can log in. User has internet to connect to the database.		
Post conditions:	User is able to open the picture from the database. User is able to play the game and is notified when they win. User stats are updated to show		
Priority:	High		
Normal Course of Events:	1. User selects a picture for the puzzle 2. User selects a puzzle size 3. The timer starts		

	4. User solves puzzle by selecting and sliding piece half of the way over another piece 5. User can opt to shuffle the board as many times as needed 6. If the vetting is successful, the Primary User will be shown a congratulatory screen with the time it took to complete the puzzle
Unsuccessful vetting result	· If user is unable to solve the puzzle there is an option to return to the previous screen to start a new puzzle
Notes and issues:	User must have an account.

5. Technology Used

5.1 List technology requirements

- The application will be coded in Java in the Android Studio environment
- The application will have to run on a phone running Android
  - Interaction will be done through touchscreen
- The app will require an internet connection
- A working camera on the phone

5.2 Additional Systems

- The application will have to interact a Loyola database
  - The pictures submitted by users creating new puzzles will be stored in the DB
  - App will have to access DB to display pictures
- The application will need an internet connection in order to connect to the database
  - Will have to be conscious of data consumption done by phone

6. Other Requirements

6.1 Performance and scalability related requirements

- It needs it work on a different sized phones
- Work at different Internet speeds

- Needs to work on all Android phones
- Server code should work efficiently when user is trying to retrieve/post picture
- Response when screen is touched should be immediate

6.2 User and usability related requirements

- They need a touch screen device
- The app is meant for casual users
  - The interface should be user-friendly, easy to learn
  - There should be a help screen for a user in case they don't understand the game
- Internet Connection
- May need to accommodate different languages (wish list)
- Possibly sound output to augment experience (click noises etc.)
  - Can be used for users who have trouble reading small text

6.3 Maintenance and portability related requirements

- Clearly commented code for adjustments/bug fixes
- Be able to handle Android updates

Plan of Development

1. Software Development Process

Once all the documentation has been completed, we plan to meet to white board a design of the app. In this session, we will discuss everything about the UI down to the buttons. Afterwards, we will assign each person in the group a portion of the app to code. To simulate a scrum type of development process, we will meet for 10 minutes on Tuesdays and Thursdays after class to discuss what each of us have accomplished and whether or not it is working. We plan to meet with the client once every two weeks to update on progress and get feedback, that way if the client wants something changed, we can update it before it gets too far into the development. Once the initial coding is done, we plan to test the app using the test cases we established. We will have the client test it and update the app with any feedback they provide. Finally, we will finish up with putting the final touches on the documentation.

2. Work Breakdown Structure (WBS)

Task	Est. Hours
Documentation (Initial requirements, SRS, Design Document, & Final Report)	35 hours
Welcome screen/log-in	5 hours
Connecting to other apps (i.e. camera/camera roll	5 hours
Gesture recognition (dragging and dropping)	15 hours

Coding Puzzle/game board logic	18 hours
Refining/Other	10 hours
Connecting database to app	11 hours
Testing	23 hours
Total	**122 hours**

3. List the development iterations and decide what will be done in each iteration?

Documentation:

- Requirements
- Use cases
- Test cases
- Whiteboard design

After this phase we will break up the work for each team member. We should all have a strong idea of how the app should function/look.

Coding:

- Welcome screen/log in
- New user screen
- Figure out how to connect to camera and other apps
- Create a database
- Add to database
- Learn how to work with gesture recognizers
- Create a grid layout with image view

We will have a base model app that we can now test.

Testing:

- Run test cases
- Get feedback from client
- Update app
- Re-test

We will have the final product.

4. Estimate the required effort (in terms of man-hour)

- Documentation- 35 hours
    - Jessa: Use cases
    - Ellen/Arturo: Requirements
    - Ellen: Features
    - Kersley: Technology used/other requirements
    - Jessa: Development plan

- Coding breakdown- 60 hours
    - Arturo: Welcome screen and log-in- 5 hours
    - Ellen: Connecting to other apps on the phone- 5 hours
    - Jessa/Ellen: Gesture recognizing- 15 hours
    - Jessa/ Kersley: Coding puzzle structure- 18 hours
    - Ellen: Creating and working the database- 7 hours
    - Arturo: Other/refining- 10 hours

- Testing- 24 hours
    - Arturo: Run test cases
    - Ellen and Jessa: meet with client
    - All: Updating app
    - Kersley: Re-test

4.1 Develop a simple table of your task list, relationships amongst the tasks, and their required time to complete.

Task List	Relationship	Time to complete (hours)
Documentation	Required to develop cases for testing and the design will serve as a blueprint for coding	35
Coding	Follows requirements and design	60
Testing	Use test cases to make sure code works correctly	24