Contents

1 Project 1.A: Music Player

TA Assigned: Riselda Kodra[1]

Objective: Design an app able to reproduce music, create and share playlists and be controlled from a smartwatch.

Material required: Tablet/Smartphone, Smartwatch.

Minimum requirements:

- The app can play audio files, showing their metadata. From the tablet, the user can pause, go to the next and previous song, and move within the audio file. Once the song finishes, the next one (if it exists) is played.
- The user can create song playlists.
- The players can create an account. They can store playlists in the cloud and share them with other users.
- When playing a file, the smartwatch displays its title, and buttons to pause and go to the previous and next songs. Also, it can recognize a shake of the wrist, which also goes to the next song.

Optional features:

- Random playing mode.
- Enable equalization when playing music.
- Recognition of additional wrist gestures.
- Turn on smartwatch screen automatically when the song changes.
- Explore the use of APIs such as SoundCloud.

Useful APIs for Media Playing and Audio Files

MediaPlayer[2], AudioTrack[3], Equalizer[4], SoundCloud[5]

[1] riselda.kodra@epfl.ch
[2] https://developer.android.com/reference/android/media/MediaPlayer
[3] https://developer.android.com/reference/android/media/AudioTrack
[4] https://developer.android.com/reference/android/media/audiofx/Equalizer#setBandLevel
[5] https://developers.soundcloud.com/docs/api/guide

2 Project 1.B: Theremin App

TA Assigned: Riselda Kodra[1]

Objective: Develop an app emulating a *theremin-like instrument*, in which the movement of the hands will determine the volume and pitch of the sounds produced.

Material required: Tablet/Smartphone, two Smartwatches

Minimum requirements:

- The user is able to produce music with the movement of their hands. By wearing one smartwatch in each hand, their movements can be recorded via the accelerometer and gyroscope. The obtained data is then sent to the tablet to be interpreted, and is translated into sound. The produced sound has two main components, pitch and volume. Each one is dependent on the movement of the corresponding hand: the faster the movement, the higher pitch or volume.
- The users have registered accounts and can add other users as friends.
- The users can choose ranges of pitch and volume for an interpretation, and store these configurations into their account and share them.
- The app allows the user to record melodies and share them with other users.

Optional feature:

- Make the pitch and volume dependent on the position of the hands instead of their speed, in a more similar way to a proper theremin. This needs a calibration of the initial position, and an algorithm for converting accelerometer/gyroscope data into distance or rotation measurements.

Useful APIs for using Smartwatches as Gestures Detectors

Using two smartwatches with the same tablet[2], Motion Sensors[3], Wrist Gestures[4], Gesture Recognition[5]

[1] riselda.kodra@epfl.ch
[2] https://developer.android.com/training/wearables/data/sync
[3] https://developer.android.com/develop/sensors-and-location/sensors/sensors_motion
[4] https://developer.android.com/training/wearables/user-input/wrist-gestures
[5] https://github.com/MLGBand/MLGBand

3 Project 1.C: Synthesizer App

TA Assigned: Riselda Kodra[1]

Objective: Develop an app where users can play and record sounds of their own design.

Material required: Tablet/Smartphone, Smartwatch

Minimum requirements:

- Allow the user to configure a sound, using one of four basic waveforms (sine, triangle, sawtooth and square waves) and a tunable filter (lowpass, bandpass, etc.). Once the sound is configured, it can be played with different pitches via buttons or a keyboard.
- While playing the sounds, the smartwatch monitors flicks of wrist to perform an action. E.g., a flick can change the sound configuration being played or its octave.
- The users have registered accounts and can add other users as friends.
- The app allows the user to record melodies and share them with other users.

Optional features:

- Add new sound configuration posibilities. For example, new waveforms, or envelopes.
- The screen of the smartwatch shows a summary of the current sound configuration.
- Add new smartwatch shortcuts through different wrist gestures. For example, allow to bend the pitch according to the movement of the wrist.
- The users can store sound configurations and share them with their friends.
- The user can store sound samples, or record them with the microphone, and play them as a soundboard.
- Enable a loop station mode, where the user can record a loop and then use it as accompaniment for further music playing.

Useful APIs for Media Playing and Audio Files

MediaPlayer[2], AudioTrack[3], Equalizer[4]

[1] riselda.kodra@epfl.ch
[2] https://developer.android.com/reference/android/media/MediaPlayer
[3] https://developer.android.com/reference/android/media/AudioTrack
[4] https://developer.android.com/reference/android/media/audiofx/Equalizer#setBandLevel

4 Project 2.A: Fitness Companion

TA Assigned: Qunyou Liu[1]

Objective: Develop an application that turns a drone into a fitness companion, supporting outdoor activities such as running, hiking, and climbing. The drone will follow the user autonomously, carry essential items, capture photos/videos, and provide real-time assistance.

Material required: Tablet/Smartphone, Smartwatch, Drone.

Minimum requirements:

- Call the API from the Parrot drone to control the movement.
- The drone autonomously tracks the user's movements during the workout, adjusting its position to stay aligned.
- The drone can capture photos and videos during the workout.
- Each user's captured media is stored in a gallery that syncs with cloud storage (e.g., Firebase) for easy access and backup.
- A live video feed from the drone's camera should be available on a connected tablet or smartphone.
- The application must include safety features such as an emergency landing triggered by a low battery or loss of connection.

Optional feature:

- Allow users to manually control the drone during the session if desired.
- Implement an inspection mode where the drone can be sent to a specific location on a map, capture video/images, and return, aiding in path safety assessments.
- Enable drone commands via hand gestures detected by the smartwatch's motion sensors.
- Introduce a coaching mode where the drone sets the workout pace, adjusting based on the user's heart rate as monitored by the smartwatch.

[1] mailto:qunyou.liu@epfl.ch

- Enhance the drone's ability to autonomously position itself for optimal photo and video capture during different exercises.

5 Project 2.B: Aerial-Selfie-Drone

TA Assigned: Qunyou Liu[1]

Objective: Develop an application that allows users to capture dynamic aerial selfies, or ”dronies,” using a drone. The app will offer customizable settings, enabling the drone to autonomously position itself for optimal shots while providing users with various creative options.

Material required: Tablet/Smartphone, Drone.

Minimum requirements:

- Call the API from the Parrot drone to control the movement.
- The drone should autonomously adjust its position based on the user’s GPS location, ensuring the best angle for capturing ”dronies.”
- Users can configure parameters such as the capture area, angle, media type (photo or video), and video duration.
- A manual ”dronie” mode allows the user to view the drone’s live feed, adjust its position, and manually capture the perfect shot.
- The drone should return to its initial position after capturing the ”dronie.”
- Upon landing, the application should automatically transfer all ”dronies” to the connected device (Tablet/Smartphone) to free up drone storage.
- The application must include safety features, such as emergency landing triggered by low battery or loss of connection.

Optional features:

- Automatically store all ”dronies” in a cloud database (e.g., Firebase) whenever an internet connection is available and share them with friends.
- Introduce a burst mode where the drone captures multiple images while moving through different angles, creating dynamic aerial effects.
- Enable a 360° video recording mode that captures a full panoramic view.

[1] mailto:qunyou.liu@epfl.ch

6 Project 2.C: Drone Surveyor

TA Assigned: Qunyou Liu[1]

Objective: Build an Android app that lets a pilot define a land area and desired ground resolution (meters covered by each photo). The app overlays a capture grid and automatically triggers the drone camera to take photos that cover all grid cells. After flight, the app combines the photos into a single mosaic image.

Material required: Tablet/Smartphone, Drone.

Minimum requirements:

- Call the API from the Parrot drone to control the movement.
- Users draw or edit a polygon Area of Interest (AOI) on a map and set a tile size (meters per picture).
- The app overlays a grid that fully covers the AOI based on the chosen tile size and shows how many cells/photos are needed.
- The app computes a recommended flight altitude from the requested ground footprint and camera Field of View, and validates altitude during flight.
- The app plans a coverage path over the grid and shows a preview before launch.
- The drone executes the path autonomously while the app automatically triggers the camera at each grid cell; captured cells are marked as done.
- On completion the drone returns to home. After landing, the Android app imports the photos and stitches a single mosaic image, saving it to the device.

Optional features:

- Overlap control: Let users set the required overlap between images to adjust grid spacing; support distance-lapse triggers (capture every D meters) as an alternative to per-cell triggering.
- Coverage quality assurance: Show a coverage heatmap, flag missed cells or low-quality shots (e.g., blur/exposure warnings), and suggest targeted re-flights for filling up gaps.

[1] mailto:qunyou.liu@epfl.ch

7 Project 3.A. The gARden

TA assigned: Christodoulos Kechris[1]

Objective: Create a social Augmented Reality (AR) interactive experience, starting from an exisiting AR demo app. Design your virtual garden, take care of your plants and grow your collection, compete against friends to create the most stunning AR garden.

Material required: Smartphone/Tablet, Wearable Watch

Minimum requirements:

- User profile system. The user should be able to create their own personal profile. Each user has a friends list, the user can send invitations to their friends to join their friends list.
- AR Garden design. The user can *place* their plants in their gARden, selecting the plant of their choice and then clicking on a surface to place it. The app should allow the user to modify the plant: rotate, enlarge or shrink, move it or delete it. Each plant has their own *health points* which the user should maintain. The more plants a garden has the more total points the user gets.
- Watering the plants. The plants should have a watering schedule, e.g. once every day. When the plants need to be watered, a notification should be sent to the user. The notification should pop up in the smartwatch. The use selects which plans to water. If the user misses a watering session the plant health decreases, until the user waters them. When a plant's health points falls to zero, it is removed from the AR garden. The amount of water is limited, the user should use it with caution. When the water runs out, a cool-down period begins, after which the user can water their plants again. Each plant model should have different watering schedule which should be displayed in an information panel.
- gARden map. The user should be able to see the position of their friends' gARdens a gARden map using googleMaps.

[1] christodoulos.kechris@epfl.ch

Optional features:

- Visit friends' gardens. The user can also load the garden of their friend to admire it and exchange design ideas.
- Garden competition. At the end of each week a garden competition takes place in each friend group. The user with the most total points (aggregating over all plants) in their gARden wins. A leaderboard should display the ranking of the users within each group of friends. Users cannot see the leaderboards of groups they are not part in.
- Plant watering with hand gestures. Use the smartphone accelerometer to detect the hand angle to imitate watering the plants.
- Equipment upgrades. Each user has a virtual currency which allows them to by a larger water tank or better watering sensors. This allows the user to water more plants at the same time, or get more optimised notifications on their plants' watering needs. Each user starts with a predetermined virtual budget. Each day their budget increases based on their garden points. If they win a week contest among their friends then they get an additional bonus.
- Best garden voting. The week contest now gets two prizes: the most garden points and the most popular garden. On Saturday the users can load their friends' gardens and then anonymously vote which one they consider the best. The most voted garden wins the popularity prize.
- Implement seasonal changes. The app should check the season (winter, spring, summer, autumn) and should update the watering needs and points of the plants.

Helpful resources:

- SceneView[1]: A library for developing AR Android applications using Jetpack Compose powered by Google Filament and ARCore.
- Sketchfub[2]: Find and use models compatible with SceneView library.

[1] `https://github.com/SceneView/sceneview-android`
[2] `https://sketchfab.com/feed`

8 Project 3.B. Virtual pets

TA assigned: Christodoulos Kechris[1]

Objective: Create a Augmented Reality (AR) interactive game, starting from an existing AR demo app. Take care of your virtual pet, go on long walks and socialise with your friends!

Material required: Smartphone/Tablet, Wearable Watch

Minimum requirements:

- User profile system. The user should be able to create their own personal profile. Each user has a friends list, the user can send invitations to their friends to join their friends list. When a user registers for the first time they select if they are a cat or a dog person.
- Taking care of the pet. The user needs to feed their pet at a specific schedule. A notification should be sent to the smartwatch. The user then has to open the AR app on their smartwatch and press the appropriate button for feeding. The user should also take the pet on a walk - again notified by the smartwatch. The app should track the distance traveled during the walk - the longer the walk the happier the pet!
- Socialise! Send your dog or cat to play with your friends' pets. More socialisation gets the pet happier :)
- The weekly challenge! The user with the most *happy* points in a friends group wins the weekly challenge - winning the user points. A global leaderboard displays all users' points sorted by weekly challenge points.
- Take a photo! Each user can take a photo with their pet. In the weekly challenge the users sees all the photos taken in their friends group and votes the funniest one. The most voted photo wins the popularity contest, winning the user additional user points in the leaderboard.

Optional features:

[1] christodoulos.kechris@epfl.ch

- Use animations. Find and use animal models that have animations. For each activity animate the model accordingly. Find also 3D models to use during feeding the pet. Use Sketchfub[1] to browse models with animations.
- Implement group socialising. Enable socialisation for many pets among a friends group.
- Cats and dogs are different! Implement different behaviour systems (feeding, need for taking for walks, etc.)

Helpful resources:

- SceneView[2]: A library for developing AR Android applications using Jetpack Compose powered by Google Filament and ARCore.
- Sketchfub[3]: Find and use models compatible with SceneView library.

[1] https://sketchfab.com/feed
[2] https://github.com/SceneView/sceneview-android
[3] https://sketchfab.com/feed

9 Project 4.A. Gesture Genius

TA Assigned: Amirhossein Shahbazinia[1]

Objective: Implement a Gesture-Memory game that utilizes a model for detecting hand gestures, challenging players to memorize and replicate sequences of gestures for both hands.

Material required: Smartphone/Tablet, Wearable Watch * 2

Minimum requirements:

- Gesture Learning Phase:
  - Display a sequence of hand gestures on the phone/tablet screen to memorize.
  - Include gestures for both hands, adding complexity as the game progresses.
- Gesture Mimicking Phase:
  - Utilize a deep learning model to validate the hand gestures in real-time.
  - Provide immediate feedback on the device and wearable: a 'success' or 'failure' indicator based on how accurately the gestures were replicated.
- Timing Mechanism:
  - Display a countdown timer during the gesture mimicking phase to add pressure and challenge.
  - Use visual and vibration cues to indicate when time is running out.

Optional features:

- **Cloud Sync:** Securely store scores and achievements in the cloud, enabling leaderboards and cross-device access.
- **Performance Metrics:** Monitor heart rate and gesture stability for dynamic gameplay and deeper performance insights.
- **Score Display:** Show scores and achievements on the device, enhancing player engagement with real-time notifications.

[1] amirhossein.shahbazinia@epfl.ch

10 Project 4.B. EmoVerse

TA Assigned: Yuxuan Wang[1]

Objective: Implements a real-time face landmark[2] and expression analysis app that creates customizable live avatars and supports mood tracking for both tablets and wearables.

Material required: Smartphone/Tablet, Wearable Watch * 2

Minimum requirements:

- 3D Face Landmark Detection for Expression Monitoring
  - Detect facial expressions (e.g., Happy, Neutral, Surprised) in real time, and display detected expressions simultaneously on both the tablet app and the wearable watch.
- Generate past expression data and generate visual reports (e.g., daily smile count, weekly mood trends).
- Avatar State Mapping
  - Provide a customizable live avatar whose state changes dynamically to match the user's expression. The avatar can be displayed both on the tablet and the watch.
- Snapshot Feature
  - Allow users to capture and save snapshots of interesting or memorable moments during live expression tracking.
  - Implement an algorithm that automatically detects and captures special movements performed by the user.

Optional features:

- **Cloud sync for cross-device access**: Synchronize captured snapshots and mood history to the cloud, allowing users to access data seamlessly across devices.

[1] yuxuan.wang@epfl.ch

[2] https://ai.google.dev/edge/mediapipe/solutions/vision/face_landmarker

- **Psychological expert support**: Provide an option to share emotion history and mood trends, which can be used for seeking deeper analysis and feedback.
- **Notification with daily report generation**: Generate a summary of the user's mood and expressions at the end of each day, delivered as a notification on both tablet and watch.
- **Emotion history for mood tracking**: Combine the heart rate data from the watch to support predictive modeling: forecast the user's mood for the next hour/day using past data.

11 Project 5.A. Red Light Green Light

TA Assigned: Dimitra Tatli[1]

Objective: Develop an interactive Red Light/Green Light physical activity game using mobile and wearable sensors to recognize user motion and promote reaction speed and self-control.

Material required: Smartphone/Tablet, Smartwatch

Minimum requirements:

- Motion Detection:
  - Use smartphone accelerometer and wearable sensors to detect user movement
  - Randomly alternate between “green light” (go) and “red light” (stop) phases, using sound and haptic cues.
- Rule Enforcement:
  - When “red light” is active, motion detection triggers a penalty and immediate feedback (sound/vibration).
  - Score is determined by distance covered during “green light” phases and reaction speed in stopping.
- Interface:
  - Real-time progress, score, and status displayed on the smartphone.

Optional features:

- Multiplayer Mode: Compete with friends through Bluetooth or WiFi, synchronizing game states across devices. One device is the game master, signaling the red/green light, and the rest connect to it, following its instructions.
- Cloud Leaderboards: Sync scores to a cloud service, enabling rankings and statistics.
- Health Integration: Log physical activity data to Google Fit.

[1] dimitra.tatli@epfl.ch

- Music Integration: Whenever the music plays is green light and when it stops, it is red light.

12 Project 5.B. Simon Says

TA Assigned: Dimitra Tatli[1]

Objective: Implement a digital version of ”Simon Says,” using wearable sensors and phone input to challenge users to perform actions—move, tap, jump— only when ”Simon says,” promoting attention and fast reaction.

- if someone follows an instruction that doesn’t come after ”Simon says”, points are reducted.

Material required: Smartphone, Smartwatch

Minimum requirements:

- Command Sequence:
  - App issues commands (e.g., ”raise your arm,” ”jump,” ”spin around”) via phone/voice/haptics.
  - Only movements issued with ”Simon says” should be followed.
- Algorithm Development:
  - Train simple Machine Learning models for command recognition, adapting Human Activity Recognition algorithms from the literature.
  - Wearable detects if the correct gesture was performed (using accelerometer/gyroscope data) and if false moves are made on invalid cues.
- Feedback System:
  - Real-time feedback for correct or false moves, with a scoring system based on attention and compliance speed.

Optional features:

[1] dimitra.tatli@epfl.ch

- Multiplayer Mode: Compete remotely or locally with friends in real-time Simon Says tournaments. One device serves as host of the game, so that all the rest connect to the same node.
- Mode Variations: Include visual-only or haptic-only command delivery for sensory diversity.
- Streaks/Achievements: Unlock badges for consecutive correct streaks.

13 Project 5.C. Daily Steps Duel

TA Assigned: Dimitra Tatli[1]

Objective: Design a competitive social fitness app where users "duel" friends in step-count challenges using smartphone and wearable step sensors.

Material required: Smartphone, Smartwatch

Minimum requirements:

- Step Tracking:
  - Aggregate step counts from both phone and watch throughout the day.
  - Challenge system to invite friends to a custom step-count competition.
  - Calculate metrics like step gait or speed using GPS signals.
- Competition Mechanism:
  - Real-time updates and motivational notifications during the challenge.
  - Track wins/losses and streaks within the app.
- Results Presentation:
  - End-of-day results with detailed breakdown and celebration feedback for the winner.
  - Show route throughout the day from GPS signals.

Optional features:

- Cloud Features: Sync history and leaderboards for competitive tracking.
- Handicap Mode: Adjust daily goals for participants with differing fitness baselines.
- Group Challenges: Enable group competitions (teams or free-for-all) for larger engagement.

[1] dimitra.tatli@epfl.ch

14 Project 5.D. Workout Partner

TA Assigned: Dimitra Tatli[1]

Objective: Implement a partner workout app that automatically counts exercise repetitions (push-ups, squats, jumps, etc.) and recognizes exercises from motion data using ML, so users can train together and challenge each other.

Material required: Smartphone, Smartwatch

Minimum requirements:

- Rep Counting:
  - Use accelerometer/gyroscope signals from phone or wearable to segment individual repetitions.
  - ML model (optionally pretrained, e.g., SVM, CNN, kNN) detects peaks/valleys in key features for each exercise.
- Exercise Classification:
  - Recognize exercise type (e.g., push-up, squat, curl).

Optional features:

- Feedback: Intensity scoring (speed, amplitude, range of motion) and cheat detection.
- Streaks and Achievements: Keep users motivated with gamification.
- Suggestions: Propose workouts to users depending on their fitness level.
- Add GPS tracking and show workout route on maps.

Helpful ML resources:

- GitHub: Fitness Tracker with ML-based rep and exercise detection using inertial signals
- tutorial, code, web demo; supports customization and extensibility.[2]

[1] dimitra.tatli@epfl.ch

[2] https://github.com/Veto2922/Fitness-tracker-based-on-ML-2