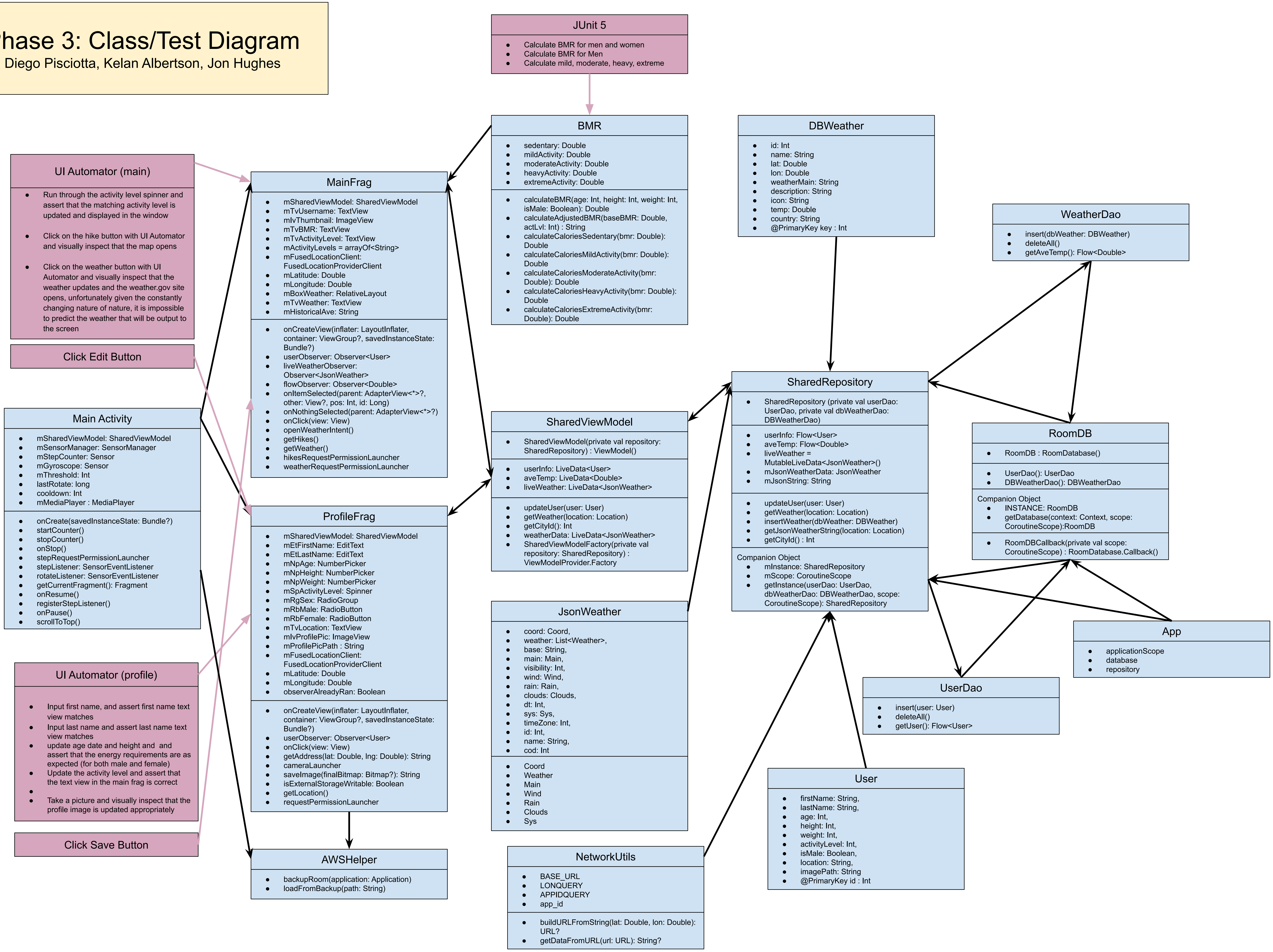
Phase 3: Class/Test Diagram

Diego Pisciotta, Kelan Albertson, Jon Hughes



JUnit 5

- Calculate BMR for men and women
- Calculate BMR for Men
- Calculate mild, moderate, heavy, extreme

UI Automator (main)

- Run through the activity level spinner and assert that the matching activity level is updated and displayed in the window
- Click on the hike button with UI Automator and visually inspect that the map opens
- Click on the weather button with UI Automator and visually inspect that the weather updates and the weather.gov site opens, unfortunately given the constantly changing nature of nature, it is impossible to predict the weather that will be output to the screen

Click Edit Button

Main Activity

- mSharedViewModel: SharedViewModel
- mSensorManager: SensorManager
- mStepCounter: Sensor
- mGyroscope: Sensor
- mThreshold: Int
- lastRotate: long
- cooldown: Int
- mMediaPlayer : MediaPlayer

- onCreate(savedInstanceState: Bundle?)
- startCounter()
- stopCounter()
- onStop()
- stepRequestPermissionLauncher
- stepListener: SensorEventListener
- rotateListener: SensorEventListener
- getCurrentFragment(): Fragment
- onResume()
- registerStepListener()
- onPause()
- scrollToTop()

UI Automator (profile)

- Input first name, and assert first name text view matches
- Input last name and assert last name text view matches
- update age date and height and and assert that the energy requirements are as expected (for both male and female)
- Update the activity level and assert that the text view in the main frag is correct
-
- Take a picture and visually inspect that the profile image is updated appropriately

Click Save Button

MainFrag

- mSharedViewModel: SharedViewModel
- mTvUsername: TextView
- mIvThumbnail: ImageView
- mTvBMR: TextView
- mTvActivityLevel: TextView
- mActivityLevels = arrayOf<String>
- mFusedLocationClient: FusedLocationProviderClient
- mLatitude: Double
- mLongitude: Double
- mBoxWeather: RelativeLayout
- mTvWeather: TextView
- mHistoricalAve: String

- onCreateView(inflater: LayoutInflater, container: ViewGroup?, savedInstanceState: Bundle?)
- userObserver: Observer<User>
- liveWeatherObserver: Observer<JsonWeather>
- flowObserver: Observer<Double>
- onItemSelected(parent: AdapterView<*>?, other: View?, pos: Int, id: Long)
- onNothingSelected(parent: AdapterView<*>?)
- onClick(view: View)
- openWeatherIntent()
- getHikes()
- getWeather()
- hikesRequestPermissionLauncher
- weatherRequestPermissionLauncher

ProfileFrag

- mSharedViewModel: SharedViewModel
- mEtFirstName: EditText
- mEtLastName: EditText
- mNpAge: NumberPicker
- mNpHeight: NumberPicker
- mNpWeight: NumberPicker
- mSpActivityLevel: Spinner
- mRgSex: RadioGroup
- mRbMale: RadioButton
- mRbFemale: RadioButton
- mTvLocation: TextView
- mIvProfilePic: ImageView
- mProfilePicPath : String
- mFusedLocationClient: FusedLocationProviderClient
- mLatitude: Double
- mLongitude: Double
- observerAlreadyRan: Boolean

- onCreateView(inflater: LayoutInflater, container: ViewGroup?, savedInstanceState: Bundle?)
- userObserver: Observer<User>
- onClick(view: View)
- getAddress(lat: Double, lng: Double): String
- cameraLauncher
- saveImage(finalBitmap: Bitmap?): String
- isExternalStorageWritable: Boolean
- getLocation()
- requestPermissionLauncher

AWSHelper

- backupRoom(application: Application)
- loadFromBackup(path: String)

BMR

- sedentary: Double
- mildActivity: Double
- moderateActivity: Double
- heavyActivity: Double
- extremeActivity: Double

- calculateBMR(age: Int, height: Int, weight: Int, isMale: Boolean): Double
- calculateAdjustedBMR(baseBMR: Double, actLvl: Int) : String
- calculateCaloriesSedentary(bmr: Double): Double
- calculateCaloriesMildActivity(bmr: Double): Double
- calculateCaloriesModerateActivity(bmr: Double): Double
- calculateCaloriesHeavyActivity(bmr: Double): Double
- calculateCaloriesExtremeActivity(bmr: Double): Double

SharedViewModel

- SharedViewModel(private val repository: SharedRepository) : ViewModel()

- userInfo: LiveData<User>
- aveTemp: LiveData<Double>
- liveWeather: LiveData<JsonWeather>

- updateUser(user: User)
- getWeather(location: Location)
- getCityId(): Int
- weatherData: LiveData<JsonWeather>
- SharedViewModelFactory(private val repository: SharedRepository) : ViewModelProvider.Factory

JsonWeather

- coord: Coord,
- weather: List<Weather>,
- base: String,
- main: Main,
- visibility: Int,
- wind: Wind,
- rain: Rain,
- clouds: Clouds,
- dt: Int,
- sys: Sys,
- timeZone: Int,
- id: Int,
- name: String,
- cod: Int

- Coord
- Weather
- Main
- Wind
- Rain
- Clouds
- Sys

NetworkUtils

- BASE_URL
- LONQUERY
- APPIDQUERY
- app_id

- buildURLFromString(lat: Double, lon: Double): URL?
- getDataFromURL(url: URL): String?

DBWeather

- id: Int
- name: String
- lat: Double
- lon: Double
- weatherMain: String
- description: String
- icon: String
- temp: Double
- country: String
- @PrimaryKey key : Int

WeatherDao

- insert(dbWeather: DBWeather)
- deleteAll()
- getAveTemp(): Flow<Double>

SharedRepository

- SharedRepository (private val userDao: UserDao, private val dbWeatherDao: DBWeatherDao)

- userInfo: Flow<User>
- aveTemp: Flow<Double>
- liveWeather = MutableLiveData<JsonWeather>()
- mJsonWeatherData: JsonWeather
- mJsonString: String

- updateUser(user: User)
- getWeather(location: Location)
- insertWeather(dbWeather: DBWeather)
- getJsonWeatherString(location: Location)
- getCityId() : Int

Companion Object
- mInstance: SharedRepository
- mScope: CoroutineScope
- getInstance(userDao: UserDao, dbWeatherDao: DBWeatherDao, scope: CoroutineScope): SharedRepository

RoomDB

- RoomDB : RoomDatabase()

- UserDao(): UserDao
- DBWeatherDao(): DBWeatherDao

Companion Object
- INSTANCE: RoomDB
- getDatabase(context: Context, scope: CoroutineScope):RoomDB

- RoomDBCallback(private val scope: CoroutineScope) : RoomDatabase.Callback()

App

- applicationScope
- database
- repository

UserDao

- insert(user: User)
- deleteAll()
- getUser(): Flow<User>

User

- firstName: String,
- lastName: String,
- age: Int,
- height: Int,
- weight: Int,
- activityLevel: Int,
- isMale: Boolean,
- location: String,
- imagePath: String
- @PrimaryKey id : Int