Newcastle University

Specification Document

NewAir

Team 22

Document Information

Project Name:	NewAir		
Prepared By:	Team 22	**Preparation Date:**	09/03/2018
Email / Phone:	v.iliev@ncl.ac.uk		
Document Version No:	6	**Document Date:**	09/03/2018

Version History

Ver. No.	**Ver. Date**	**Revised By**	**Description**
1	13/12/2017	Team 22	First draft
2	01/02/2018	Team 22	Revised GUI
3	08/02/2018	Team 22	Revised Team Roles
4	15/02/2018	Team 22	Revised Functional Requirements and Modifications
5	22/02/2018	Team 22	Revised Functional Requirements
6	09/03/2018	Team 22	Revised document structure

Table of contents

1. Purpose 5
   1.1. Document 5
   1.2. Application 5
2. Background & Analysis 6
   2.1. Analysis Process 6
   2.2. Clients and Users 6
   2.3. Current Procedures 6
   2.4. Competitor Applications 7
      2.4.1. AirVisual 7
      2.4.2. Plume Air Report 8
      2.4.3. BreezoMeter 9
   2.5. Summary 9
3. Roles and Deliverables 10
   3.1. Roles 10
      3.1.1. Front-End Development 10
      3.1.2. Back-End Development 11
   3.2. Deliverables 12
4. Project Plan 13
5. Hardware and software platforms 14
   5.1. Software 14
   5.2. Hardware 14
6. References 14
   6.1. Project overview 14
   6.2. Applications research 14
   6.3. Case studies 16

7. Solution requirements 17
7.1. Functional Requirements 17
7.2. Non-Functional Requirements 21
8. Other considerations 22
8.1. Assumptions 22
8.2. Constraints and Dependencies 22
9. Software Design 23
9.1. Overview 23
9.2. Modifications 24
9.3. System Architecture 25
9.4. Functionalities 26
9.4.1. Deployment Diagram 26
9.4.2. Component Diagrams 27
9.5. Packages and Classes 29
9.5.1. Package Diagrams 29
9.5.2. Class Diagrams 31
9.6. Dynamic Behaviour 33
9.6.1. Sequence Diagrams 33
9.6.2. Activity Diagrams 34
9.7. Graphical User Interfaces 35
9.7.1. Home 35
9.7.2. Map 38
9.7.3. History 39
9.7.4. Settings 40
10. Definition of terms 41

1. Purpose

1.1. Document

This document outlines the objectives, functionalities, development plan and implementation of the mobile application NewAir. Briefly covered are research into similar systems and development team structure.

1.2. Application

NewAir is a mobile solution providing interactive, accurate and real-time data analysis of the air pollution levels in Newcastle upon Tyne. By collecting air quality data from sensors deployed across the city by the Urban Observatory, and presenting it in various forms of interactivity and complexity, users can make informed health decisions in situations of various constraint with guaranteed data accuracy.

The goal is to provide a dependable, easy to use and versatile air pollution database, helping citizens stay mindful of pollution levels in their surroundings whilst maintaining a healthy lifestyle.

2. Background & Analysis

2.1. Analysis Process

To analyse current trends and tendencies, research was done on available air pollution applications. Several products were used and evaluated, the pros and cons of each were noted after objective analysis. Into consideration were also taken comments from users on online mobile markets. The positives were summarised, and was discussed whether they could be achieved with the available data from the Urban Observatory. Focus was given on the functionalities provided by the Observatory, and how the extracted data could cover the widest domain area, i.e. surpass other products by utilizing the widest range of available features.

Studies were also done on national pollution-related surveys, in order to analyse what features customers would find beneficial in an application.

2.2. Clients and Users

Potential users of the system are those concerned with the environment, especially the air of Newcastle.

As the information is health-related, no particular situations stand out as especially benefitting, but it can be assumed that people with active lifestyles will use it most frequently.

2.3. Current Procedures

Currently, Newcastle citizens can get live pollution data from the website of the Urban Observatory. However, it can prove difficult to use – the vast amount of data affects its responsiveness and visual clarity.

2.4. Competitor Applications

2.4.1. AirVisual

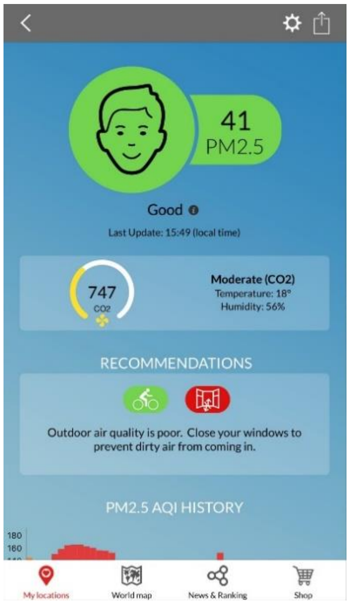


Figure 1. AirVisual

2.4.1.1. Pros

- Pleasing user interface - Hurley, B. (n. d.)
- Allows saving custom user locations – Phong, H. T. (n. d.)
- Helpful health advice – Szymborski, R. (n. d.)

2.4.1.2. Cons

- No Refresh functionality – Hurley, B. (n. d.)
- Lack of data for specific locations - Szymborski, R. (n. d.)
- No alert notifications – Magnus, W. (2018)

2.4.2. Plume Air Report

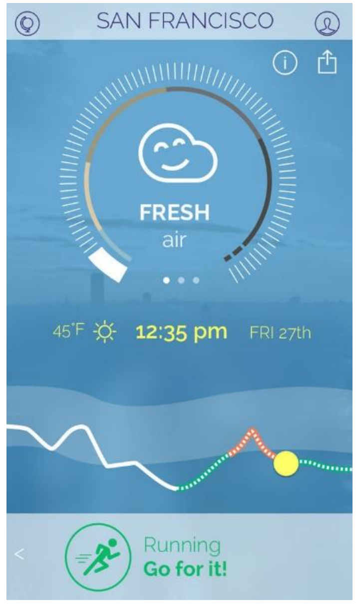


Figure 2. Plume Air Report

2.4.2.1. Pros

- Can save custom cities – Vega, L. (n. d.)
- Pleasing user interface – Vega, L. (n. d.)
- Easy to use – MacPherson, A. (2018)

2.4.2.2. Cons

- Unclear readings – Musser, E. (2018)
- Not enough data collection for particular locations – Beck, C. (2017)
- Questionable reliability for particular locations - MacPherson, A. (2018)

2.4.3. BreezoMeter

Figure 3. BreezoMeter

2.4.3.1. Pros

- Helpful health advice – Liutas (n. d.)
- Easy to use - von Schmalz, R. (n. d.)
- Pleasing user interface – Anonymous (2017)

2.4.3.2. Cons

- Doesn't specify the pollutants used to rate air quality – Ann, M. (n. d.)
- Questionable accuracy - Israeli, E. (n. d.)
- Map functionality difficult to use – Liutas (n. d.)

2.5. Summary

Popular products on the market show common traits. Information is conveyed concisely, focusing on interface simplicity and clarity (Vega, n. d.; Hurley, n. d.). Pollution data is presented informatively, but to questionable accuracy (Beck, 2017; MacPherson, 2018; Israeli, n. d.). Often it is not clear how the data is collected and what area it covers (Beck, 2017; Musser, 2018), e.g. having one sensor per city is undesirable, as pollution levels vary greatly between regions.

The data available from the Urban Observatory can be used to fill the gaps in current systems. Having access to real-time sensors across the whole city, it is possible to develop an application that provides accurate, concurrent and consistent supply of data from all regions of Newcastle, giving an objective view of air quality to all citizens in everyday activities.

3. Roles and Deliverables

3.1. Roles

3.1.1. Front-End Development

- V. Iliev

  Team Leader, acting as the main point of contact for the development team as a whole.

  Responsibilities include:

  - Delegating responsibilities among team members
  - Setting deadlines for milestones and deliverables
  - Quality Control
  - Accessibility Standards Research
  - Documentation
- V.M.T. Godsell

  Lead Front-End developer, acting as the main point of contact for the Front-End team.

  Responsibilities include:

  - Research into UX/UI Design and Development
  - Development of App Colour Palette and adjustments catering to those with colour blindness
  - Concept Design and Development of Logo and identifying images.
  - Web Development
- R.J. Cooper

  Front-End developer, working to optimize aesthetics and user experience.

  Responsibilities include:

  - Concept Design and development of background artwork
  - Research into accessibility compliance

    - UI Design
    - Web Development
    - Website Testing and QA
- A.L.J. Heilling

  Front-End developer, working to optimize UI design.

  Responsibilities include:
    - Concept and Design of In-App Sprites
    - Concept and Design of In-App Icons
    - Research into Air Quality Indexes
    - UI Design and Refinement

3.1.2. Back-End Development

- I.J. Watt

  Lead Back-End Developer, acting as the main point of contact for the Back-End development team.

  Responsibilities Include:
    - Development team sub leader
    - UML Diagram development
    - Testing of Back-End development
    - Urban observatory API research
  - Development of GPS functionality
- I. Gylaris

  Back-End developer

  Responsibilities include:
    - UML Diagram Development
    - Development of Line-Graph code
    - Testing of Back-End development
    - Formalised functional requirements specification
- L.P. Stannard

  Back-End developer

  Responsibilities include:

    - UML Diagram Development
    - Retrieval of data from the Urban Observatory
    - Testing of Back-End development.
    - Development of Google maps functionality

- B.D. Wilsher

  Back-End developer

  Responsibilities include:

    - UML Diagram Developer
    - Testing of Back-End Development
    - Development of Timer functionality

3.2. Deliverables

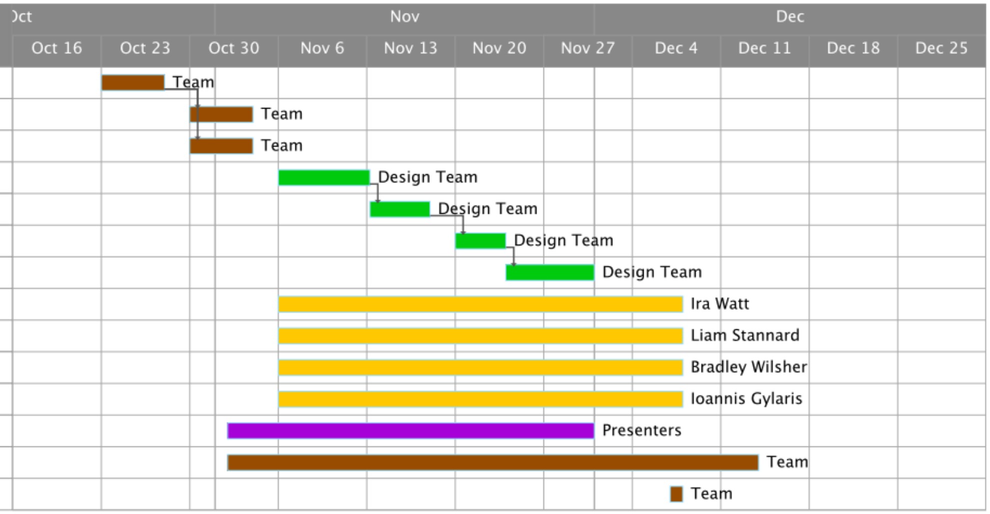
- Single Android™ APK installation file
- Product marketing website
- Product marketing poster
- Project documentation
- Project source code

4. Project Plan

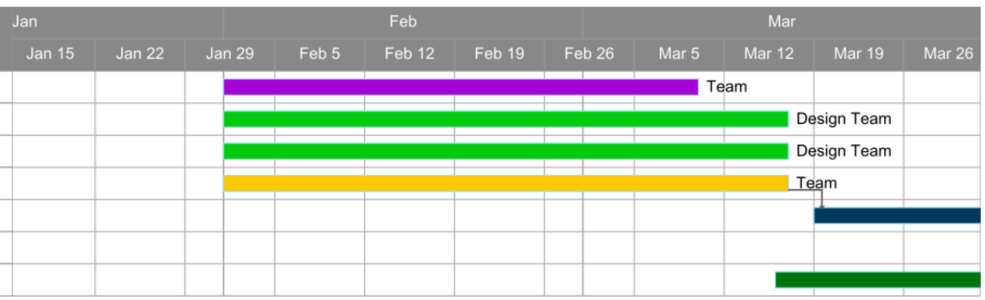
Design Team - Alexander Heiling, Rebecca Cooper, Victoria Godsell, Vladislav Iliev

Presenters - Bradley Wilsher, Liam Stannard, Vladislav Iliev

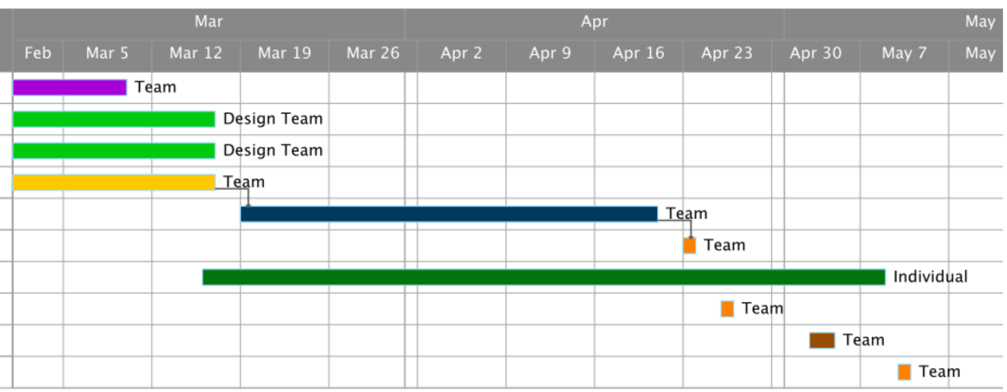
Task Name	Start	Finish	Assigned To
Team Contract	10/23/17	10/27/17	Team
Draft Project Plan	10/30/17	11/03/17	Team
Functional requirements	10/30/17	11/03/17	Team
Color scheme and fonts	11/06/17	11/13/17	Design Team
Home screen layout	11/13/17	11/17/17	Design Team
Map and Data Chart layou	11/20/17	11/23/17	Design Team
Screen mock-up	11/24/17	11/30/17	Design Team
Collect API	11/06/17	12/07/17	Ira Watt
Implement Maps	11/06/17	12/07/17	Liam Stannard
Implement GPS	11/06/17	12/07/17	Bradley Wilsher
Implement data chart	11/06/17	12/07/17	Ioannis Gylaris
Dragon's Den	11/02/17	11/30/17	Presenters
Draft Specification	11/02/17	12/13/17	Team
Peer review	12/07/17	12/07/17	Team



Task Name	Start	Finish	Assigned To
Specification	02/01/18	03/09/18	Team
Poster	02/01/18	03/16/18	Design Team
Website	02/01/18	03/16/18	Design Team
Implementation	02/01/18	03/16/18	Team
Testing	03/19/18	04/20/18	Team
Final Submission	04/23/18	04/23/18	Team
Individual Report	03/16/18	05/08/18	Individual



Task Name	Start	Finish	Assigned To
Specification	02/01/18	03/09/18	Team
Poster	02/01/18	03/16/18	Design Team
Website	02/01/18	03/16/18	Design Team
Implementation	02/01/18	03/16/18	Team
Testing	03/19/18	04/20/18	Team
Final Submission	04/23/18	04/23/18	Team
Individual Report	03/16/18	05/08/18	Individual
Trade Fair Demo	04/26/18	04/26/18	Team
Peer Review	05/03/18	05/04/18	Team
Final Tech Demo	05/10/18	05/10/18	Team



5. Hardware and software platforms

5.1. Software

- Eclipse IDE
- Android™ Studio IDE
- Android™ SDK
- Adobe Photoshop

5.2. Hardware

- Mobile device running OS Android™ 4.4 (KitKat) or newer
- Personal computer capable of running an emulated version of the abovementioned mobile device

6. References

6.1. Project overview

- Colquhoun, J. (2017) Newcastle University Stage 2 Team Project Brief [online] Available at: https://blackboard.ncl.ac.uk/bbcswebdav/pid-3366753-dt-content-rid-9621210_1/xid-9621210_1. [Accessed 9 March 2018]
- Urban Observatory (2018) *Urban Observatory Website* [online] Available at: http://www.urbanobservatory.ac.uk/.

6.2. Applications research

6.2.1. AirVisual

- Hurley, B. (2018). AirVisual. [online] Available at: https://play.google.com/store/apps/details?id=com.airvisual&reviewId=Z3A6QU9xcFRPRmd1VlVsOWtVQURTVTJoOWotM3R0eTlVTkN5M19sMVFuZ0N0cl9oV3hOWXUyRmc2Q0RxWUczMVNNSXlYLWVfNnNmQWJqX0lkQUw4eld5c0xR&hl=en_GB. [Accessed 4 March 2018].
- Magnus, W. (2018) AirVisual. [online] Available at: https://play.google.com/store/apps/details?id=com.airvisual&reviewId=Z3A6QU9xcFRPRTgtY2g2VVkzNH

VybFJJd3dWSkF3eHJSdWZxQVhKdlhvQkdiR3k2YlhsR25kOEVKUTJtbVFBSnpKTll2UjV3N29ObU5oUEtuU2toakt6TnFB&hl=en_GB [Accessed 4 March 2018].

- Phong, H. T. (n. d.). AirVisual. [online] Available at: https://play.google.com/store/apps/details?id=com.airvisual&reviewId=Z3A6QU9xcFRPR2NrUGo0TFYtNzdDRlZDZWhjbXdkWnl0Y29kVEFhZXp4X3E2R0R5UGxSQzdCUWp4NFZjVzRiUjBZUXRrc1Z5cU1QX2Y1enN6TlNNYTh1MThr&hl=en_GB. [Accessed 4 March 2018].
- Szymborski, R. (n. d.) AirVisual. [online] Available at: https://play.google.com/store/apps/details?id=com.airvisual&reviewId=Z3A6QU9xcFRPSHBKZU0yaTdjRV9feTNnTC1hcmFVWnlLNmlBQl92dFViX0hlX3FvUjFMV1psczI4RnItc1lnN29PU0E4RGcwbmhIb1FRbTdpdjJzUUtFa2lv&hl=en_GB. [Accessed 4 March 2018].

6.2.2. Plume Air Report

- Beck, C. (2017) AirVisual. [online] Available at: https://play.google.com/store/apps/details?id=com.plumelabs.air&reviewId=Z3A6QU9xcFRPSE5BTlIzT1YwOEtfNzFpUTRfcl91RzFHTC1Ba2tvSjVOLXVQbVlWU3ZLMXhkZTJzRlg0MVFjb1lUbjI3ck9pTXZsc1Z5ZXlsM19Gamo1ZHI4&hl=en_GB [Accessed 4 March 2018].
- MacPherson, A. (2018) AirVisual. [online] Available at: https://play.google.com/store/apps/details?id=com.plumelabs.air&reviewId=Z3A6QU9xcFRPRS03UklVdlpnVlEyMkNMYmktVnUzNUx1LWMzLVFOVVJOamhveC1hU3lPXzdiMTBKUU9CTUhCOG9idUNzU25maEx4anFHdU9fSGtMdmkwUnA0&hl=en_GB [Accessed 4 March 2018].
- Musser, E. (2018) AirVisual. [online] Available at: https://play.google.com/store/apps/details?id=com.plumelabs.air&reviewId=Z3A6QU9xcFRPRXRXaUVHUlM3RkdsZ2dXNktqQXlMd1dQSHU5NVhNWGJtQ01iZTk2LTdNVHZqelNZVEpsV1lPRktjalNXMlJkNVB6aWRWZkdVeFZ3aEdOLWNR&hl=en_GB [Accessed 4 March 2018].
- Vega, L. (n. d.) AirVisual. [online] Available at: https://play.google.com/store/apps/details?id=com.plumelabs.air&reviewId=Z3A6QU9xcFRPRjJrRFdZVmVyYmdKV2dnbkV1RFEySU5hTnU4QkJnUEp2cUFlX3ZTb1QxZUl0RmVPMzlNV29SSzJEeFl2OGJqdUQ2V0J1eEswOHpSNlAtMFRJ&hl=en_GB. [Accessed 4 March 2018].

6.2.3. BreezoMeter

- Ann, M. (n. d.) AirVisual. [online] Available at: https://play.google.com/store/apps/details?id=app.breezometer&reviewId=Z3A6QU9xcFRPSEpYTFZxcW83T1lxNlpzQUlsb3Fub0Jodk5XWU9Gb1NvaEFVSVlvMUhRcy1WUVlvb1FfbGdqOGQ0QzAzWlAzSXZRR1JfNUNNSnAwMGV1Rmc&hl=en_GB. [Accessed 4 March 2018].
- Anonymous. (2017) AirVisual. [online] Available at: https://play.google.com/store/apps/details?id=app.breezometer&reviewId=Z3A6QU9xcFRPRTJDNjJEMGpNV1NEY1hKOU00WUFhVUhLOEg5TlJkbWZHckQtZVRPWVQtcFBaM0lMM0lzTHpTcVlpLUI5TkVPSkVmTldaSmhlZ0t2cG53ZVE&hl=en_GB [Accessed 4 March 2018].

- Israeli, E. (n. d.) AirVisual. [online] Available at: https://play.google.com/store/apps/details?id=app.breezometer&reviewId=Z3A6QU9xcFRPR2dhMWZVU0FxRjN6S2lRaF90NTZZM3pkVFJmeG56c0dkQTJFaTFrZHVtWDBqNTJvZlZGVVN1ZEVfZ1lIbnZUQVUxUTdrc0tPWWFoVHo2bFE&hl=en_GB. [Accessed 4 March 2018].
- Liutas (n. d.) AirVisual. [online] Available at: https://play.google.com/store/apps/details?id=app.breezometer&reviewId=Z3A6QU9xcFRPR2ZDR285QlRKWDRNdno5bzhyMVJSeGd3S1NpcEt5dVZfNHBHZjlRZDk0UE1MU0ZkVWVNckZVcklCSFJUSE8wNlE3THItRkRxU3ZfbWxCcHc&hl=en_GB. [Accessed 4 March 2018].
- Von Schmalz, R. (n. d.) AirVisual. [online] Available at: https://play.google.com/store/apps/details?id=app.breezometer&reviewId=Z3A6QU9xcFRPSHRhQ0w4U0F6RGE3SDRlM2FWU21wR2M3M3FzMXpjdHhKTG9IbWFTWkdBVDlhWlc1TFM0Z0JBSHlick9DQTg2eTlrWHd5WFVPZW5ETDRHd0E&hl=en_GB. [Accessed 4 March 2018].

6.3. Case studies

- Department for Environment, Food & Rural Affairs. UK-AIR: Air Information Resource [online] Available at: https://uk-air.defra.gov.uk/ [Accessed 10 Nov 2017].
- Newcastle City Council (2017). ‘Managing and improving air quality in Newcastle’ [online] Available at: https://www.newcastle.gov.uk/environment-and-waste/pollution/air-pollution/managing-and-improving-air-quality-newcastle [Accessed 10 Nov 2017].
- Kelly, M. (2017). ‘Newcastle has some of the worst air pollution in the UK - so why do so few people care?’ [online] Available at: http://www.chroniclelive.co.uk/news/north-east-news/newcastle-worst-air-pollution-uk-12676181 [Accessed 10 Nov 2017].
- Jawad, S. (2015). ‘Mapped: Where is air pollution killing the most people?’ [online] Available at: http://www.telegraph.co.uk/news/earth/environment/11991350/Mapped-Where-is-air-pollution-killing-the-most-people.html [Accessed 10 Nov 2017].

7. Solution requirements

7.1. Functional Requirements

Requirement	Priority (H, M, L)	Supplier Compliance (Full, partial or will not be delivered)	Supplier Comment
1. General			
1.1. Installable on devices running OS Android™ 4.4 (KitKat) or higher	H	Full	The ability of the user to install the app on his/her device
1.2. Connect to Urban Observatory via RESTful API	H	Full	In order to collect live and past data regarding PM10,PM2.5,O3, temperature humidity readings
1.3. Locate sensors collecting data on pollution, temperature and humidity	H	Full	Find all available sensors to retrieve data from
1.4. Extract sensor data in JSON format	H	Full	Data to be shown on screens
1.5. Extract sensor coordinates	H	Full	This will be later used to display the correct location of each circle on the map screen
1.6. Collect past data from Urban Observatory	H	Full	Every day up to a week old from the current date.
1.7. Locate user coordinates via GPS	H	Full	To be used for requirement 1.8

1.8. Locate nearest sensors to user coordinates	H	Full	This will allow the app to display the air pollution level nearest to the user and also display that sensor's distance from the user's location
1.9. Collect pollution, temperature and humidity levels for a set of locations	H	Full	This data will be displayed on the home screen
1.10. Have a consistent navigation bar to switch between display screens	H	Full	This will allow the user to navigate between the different screens of the app
1.11. Present loading screen while program components load	H	Full	This will let the user know that background processes setting up the app's functionality are running at that moment
2. Home			
2.1. Provide air pollution, temperature and humidity for a set of locations	H	Full	Displaying a nearby location by default and then using requirement 2.4 in order to rotate between custom user locations. Doing this changes the data displayed according to the selected location
2.2. Offer adding custom user locations	H	Full	Allows the user to select a location from the map for which he/she would like to receive information for
2.3. Have a carousel to switch between custom locations	H	Full	A flexible way of allowing the user to select which area's

			information is currently displayed
2.4. Provide manual Refresh button	H	Full	Fetches the most recent data from the Urban Observatory
2.5. Provide access to Settings	H	Full	Allowing the user to change different aspects of the app, for more detailed information look at section 4 of this table
2.6. Have a changing background colour	H	Full	This will provide the user with a visual feedback about the current levels of pollution
3. Map			
3.1. Display sensor readings as coloured areas	H	Full	This will show all available sensors within Newcastle on the map
3.2. Provide manual Refresh button	H	Full	Fetches the most recent data from the Urban Observatory
3.3. Each circle on the map should have a colour displaying its current pollution readings	H	Full	This provides a visual feedback to the user about the current pollution level at the area of the sensor
3.4. Provide a Legend button	H	Full	Provides information about each colour a circle can have on the map
4. History			
4.1. Display line graph on overall pollution for the city	L	Full	Displays pollution levels for Newcastle over the course of a week in the past

4.2. Interactive graph, pressing a timestamp updates the pollutant levels accordingly	L	Full	Showing the specific pollution values recorded at the date selected
4.3. Provide manual Refresh button	L	Full	Dynamically fetches past data (a week) from the Urban Observatory
5. Settings			
5.1. Toggle colour blind mode	H	Full	Change the colour palette to one that is suitable for colour blind people
5.2. Remove custom user locations	H	Full	Remove one or all of the custom locations the user has added
5.3. Change data auto update interval	H	Full	The user can select an interval from 15s up to 11 minutes by enabling the automatic updates (by default new data is retrieved by manually pressing the refresh buttons at each screen)
5.4. Display developer information	M	Full	“About” section containing information about the team

7.2. Non-Functional Requirements

Requirement	Priority (H, M, L)	Supplier Compliance (Full, partial or will not be delivered)	Supplier Comment
1. Home screen background colour should comply with government-regulated air pollution colour scheme	H	Full	Consistent UK-wide air quality recognition
2. Home screen background colours should comply with the WCAG 2.0 Accessibility Standard	H	Full	Accessibility for users with visual impairments
3. Program navigation should comply with Google Android™ Navigational standards	H	Full	Consistent navigational experience
4. Navigational bar should be consistent across all screens	H	Full	Non-distracting changes of state
5. Colours of Map sensors should have a contrast ratio over 5.00:1 to stand out from underlying map	H	Full	Visual recognition of program functions

8. Other considerations

8.1. Assumptions

- All mobile devices have Internet connectivity
- All mobile devices have GPS connectivity
- All mobile devices have processing power to run the application

8.2. Constraints and Dependencies

NewAir supports devices running OS Android™ 4.4 (KitKat) or newer (released sooner than October 31st, 2013).

The software is dependent on multiple external services, namely the Urban Observatory and their sensors, Internet connectivity, and GPS (in order to fetch the closes sensors in user proximity). In cases they fail, software accuracy is not guaranteed:

1. Sensor data is downloaded as is - in case data is inaccurate and/or inefficient, there are no means of verification or correction.
2. If a connection to the sensor database cannot be established (i.e. Internet connectivity is unavailable, or Urban Observatory seizes operation), the user is notified on program start-up.
3. If GPS connectivity is unavailable, NewAir retains core functionality, disabling only the respective nearest sensor fetching functions.
4. The accuracy of the GPS connectivity cannot be verified. In any case, the user is presented with the distance to the nearest sensors via the Home screen.

9. Software Design

9.1. Overview

NewAir presents air quality data across multiple functionalities, each varying by simplicity, interactivity and depth of information. Users choose which feature to use depending on their level of interest and desired area of sensor coverage:

1. Home – provides simplified summary of air quality in the proximity of the user, for the city as a whole, and any user-added custom locations. With a simplistic user interface, it gives a quick snapshot of the pollution levels when in-depth analysis is not required.
   Core functionality is a colour-changing background, offering immediate recognition of pollution levels by adapting its colour palette to government-regulated air quality colour codes. Colour blind palette is offered for users with visual impairments.
2. Map – offers visual representation of pollution levels across the city by providing an interactive colour-coded map. Air pollution sensors are displayed as circles, filled with colour according to the aforementioned standardized pollution colour palette.
3. History – line graph, representing daily pollution levels over a weeks' time. Complemented with a breakdown of air pollutants. Available for overall city pollution. Used for in-depth exploration of past air quality levels.

To provide real-time data, NewAir fetches updated sensor information in predefined time intervals, or manually, when the user desires. Refresh time intervals can be user-modified. That way, intermediate customer requirement of updating data is covered across the whole system.

NewAir will not provide pollution data beyond the scope of the Urban Observatory. Only Live and History data (for the past week) is of interest, future pollution forecasts cannot be made on basis of the available statistics.

9.2. Modifications

During prototype development, user reviews were collected on the user interface and experience. Development modifications reflected:

- Colour palette (improving accessibility for users with visual impairment)
- Positioning of interface elements (ease of navigation)
- Core functionalities

Resulting modifications included:

- Regulating colour choices – using standardized colour palette
- Redesigning navigational features – using standardized Google navigation methods (navigational bar methodologies)
- Adding option for saving custom user locations
- Adding additional customization settings – enabling colour blind palette
- Simplifying functionalities – several features could not be implemented due to constraints on Android Google Map.

9.3. System Architecture

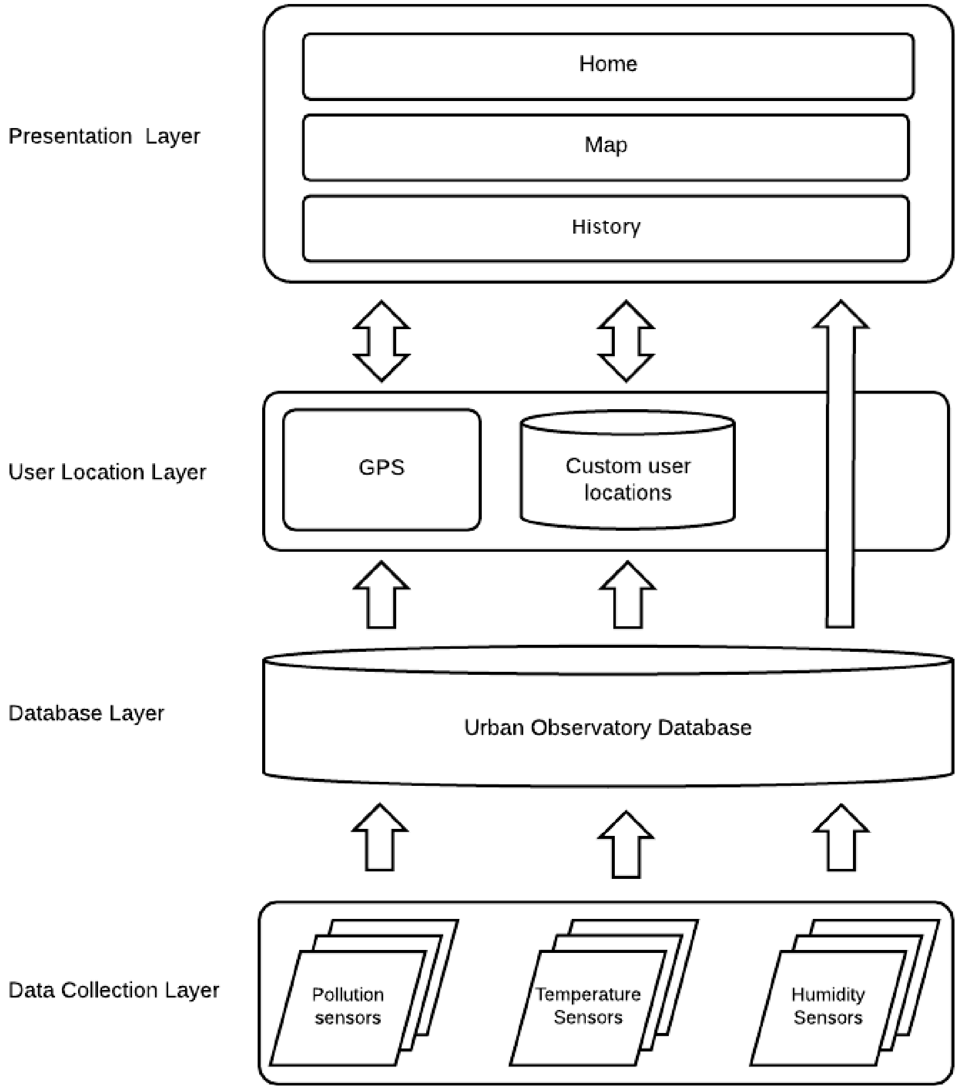


Figure 4. NewAir System Architecture

The foundation of NewAir lies in the collected sensor data, which propagates through the Urban Observatory Database to the user interface. NewAir establishes connection to the Database through extracting pollution, temperature and humidity information for specific locations – current user location (if GPS is enabled), custom user-added locations (if such are added), and all city sensors. Data is then presented according to the currently used functionality.

The User Location Layer illustrated above covers Basic, Intermediate and Advanced functionalities – it enables the user to add and navigate between saved locations of choice, as well as provide exceptionally relevant sensor data by offering air quality from nearest sensors.

9.4. Functionalities

9.4.1. Deployment Diagram

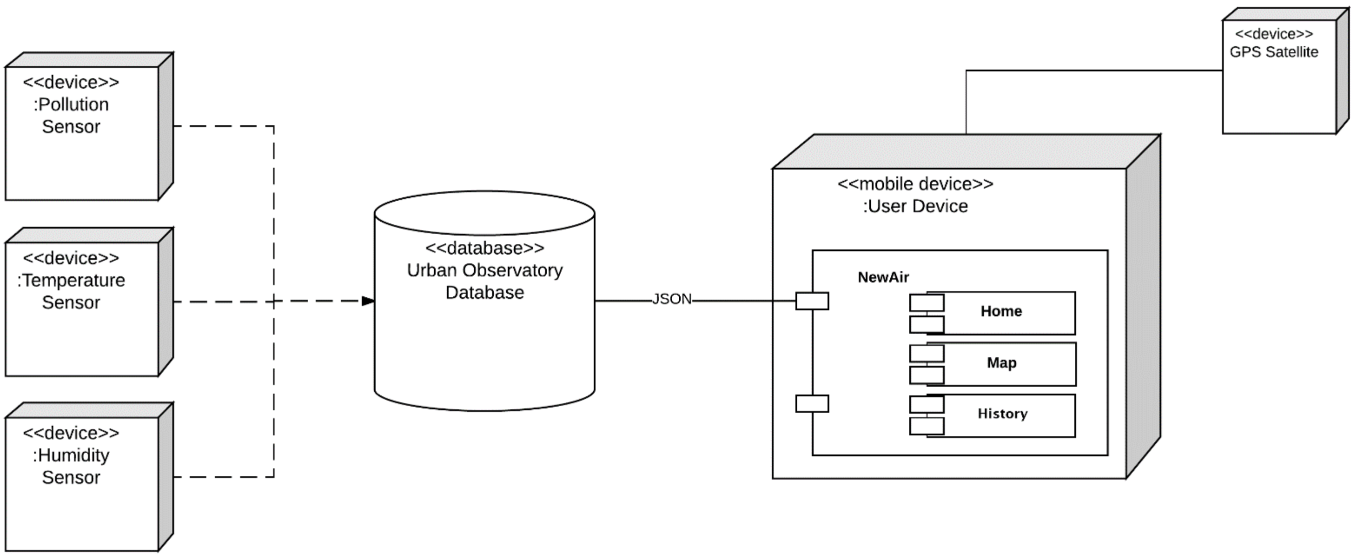


Figure 5. NewAir Deployment

In order to provide full functionality, the deployed software system should have access to an air quality database (in this case, data is extracted from the Urban Observatory Database through JSON format), which in turn should have reliable connection to pollution, temperature and humidity sensors.

GPS connectivity is optional, but not imperative – if such is not available, NewAir cannot provide data from the nearest sensors. All other functionalities are retained.

9.4.2. Component Diagrams

In order to provide a concise, but informative overview of components, focus will be given on the two most fundamental aspects – the Home screen and its data (where the user is expected to spend most time), and the persistent storage system (where NewAir records custom user locations and history data).

9.4.2.1. Home Data

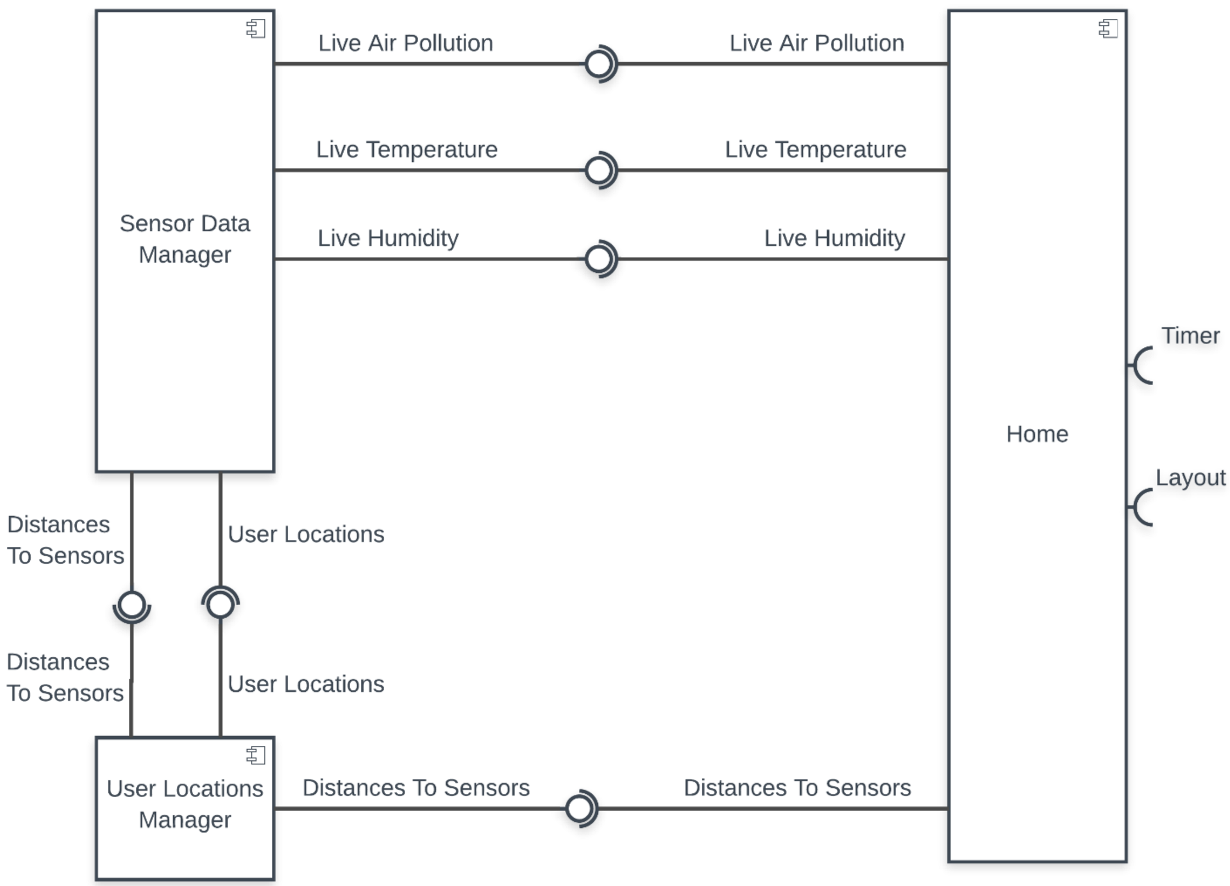


Figure 6. Home Data Components Diagram

To provide data on pollution, temperature and humidity, data is fetched through a dedicated Sensor Data Manager component – it downloads, parses and distributes all sensor readings.
The Sensor Data Manager transfers all live data to the Home carousel. Because it holds all information for available sensors, the Manager also calculates distances between user locations (including the GPS position) and their nearest sensors. Therefore, a User Locations Manager

component must provide the coordinates of the user locations, and then store the distances to be displayed in Home.

As an additional feature, Home (and all other Fragments) share a Timer, which automatically downloads data at predefined time intervals. Even though Home does not interfere with data updates, it needs to be connected to the Timer, because if a user manually presses Refresh, the Timer needs to be reset.

9.4.2.2. Persistent Data Storage

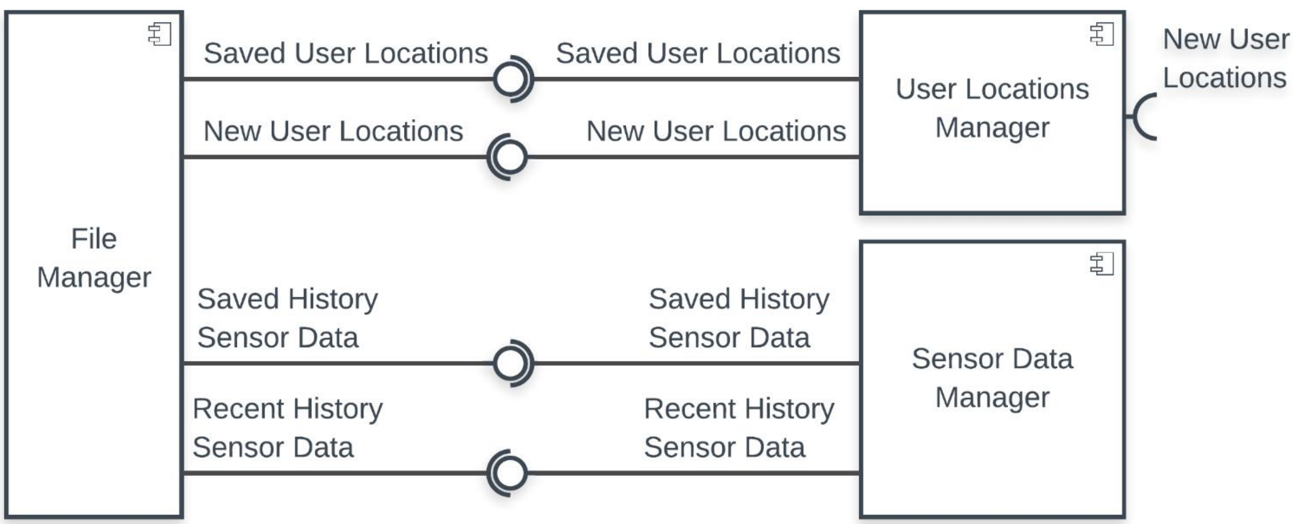


Figure 7. Data Storage Components Diagram

A specialised File Manager component carries out the reading/writing of data in device memory. NewAir makes it possible for Users to store map coordinates of their choice, to extract sensor data most relevant to preferred locations. In order to allow persistent storage of said locations, their coordinates and names are managed during runtime by the dedicated User Locations Manager, and are stored in phone memory by the File Manager upon edit.

NewAir allows viewing past pollution data over the course of a week. In order to conserve mobile data bandwidth, all sensor data for past days is stored in phone memory (it is assumed past data will not change, so it does not need to be downloaded multiple times).

9.5. Packages and Classes

9.5.1. Package Diagrams

Focus will be given on the overall program layout, notably code structure and sensor data structure.

9.5.1.1. Program structure

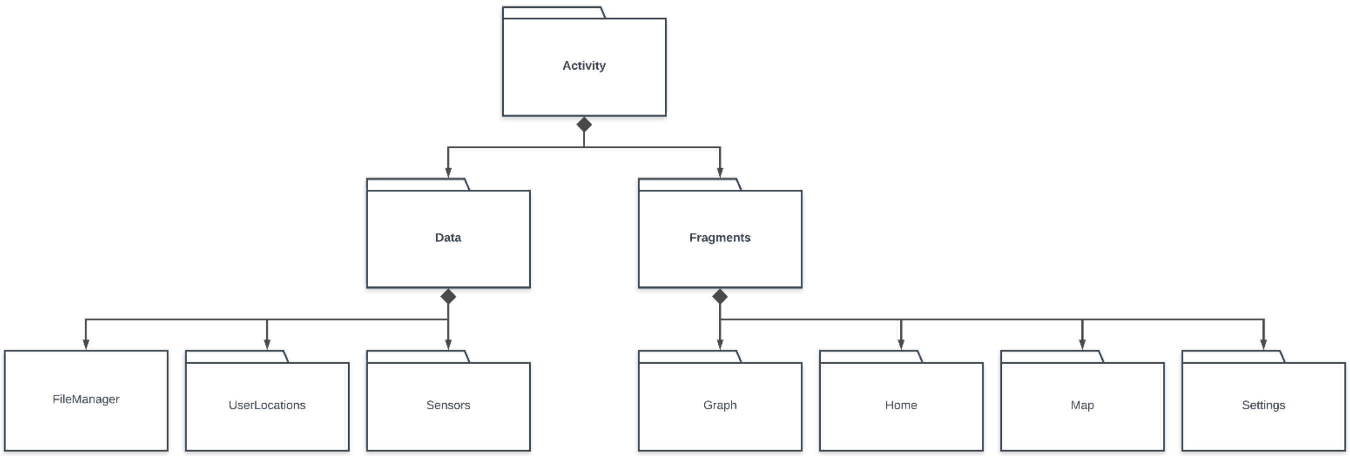


Figure 8. NewAir package structure

NewAir emphasises clear separation between data management and UI.
Independence is achieved by splitting data classes and View classes. Further decoupling both categories into respective subgroups (i.e. Persistent Storage, User Locations and Sensors for the Data, and respectively all GUI Fragments) achieves cohesive, but versatile system structure, following standard mobile development guidelines.

9.5.1.2. Sensor data

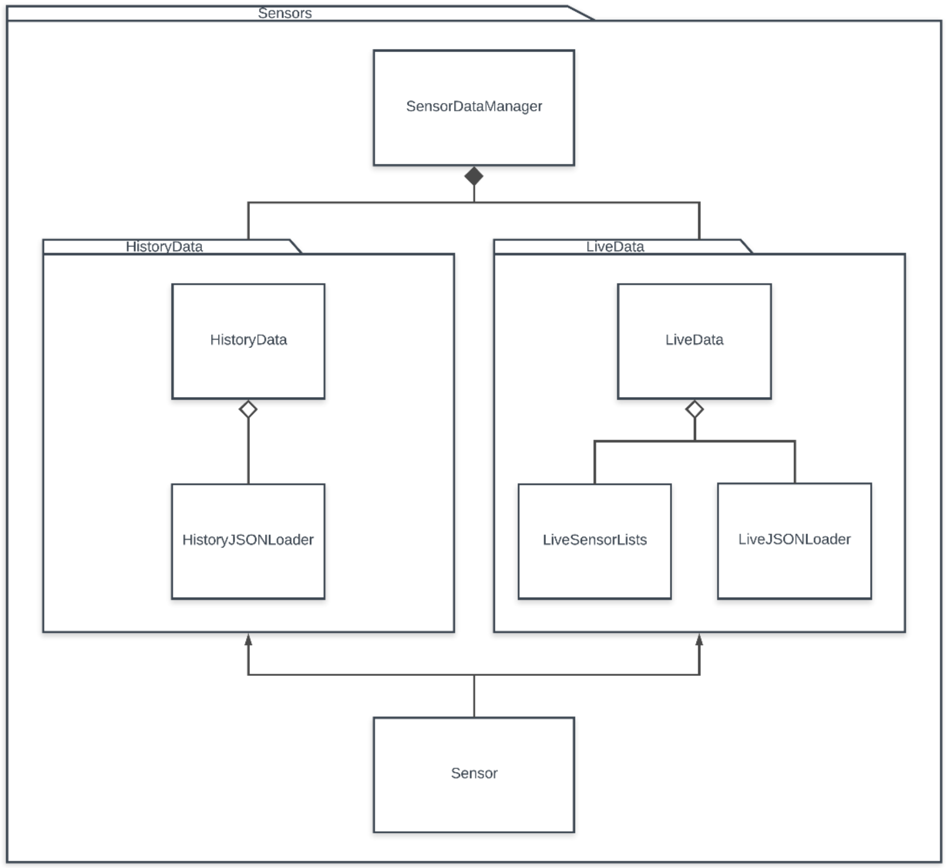


Figure 9. Sensor Data Management Structure

The dedicated Sensor Data Manager allows gathering both Live and History data.
Once downloaded, History data requires minimal maintenance. A simple pair of storage and download class ensure proper management and transfer between components.
Live data offers more complex functionalities, e.g. calculating distances between map coordinates, and so requires further separation between storage, download and calculating functions.
Both data components make use of the general Sensor class, allowing for clear implementation of sensor types and coordinates.

9.5.2. Class Diagrams

9.5.2.1. Sensor

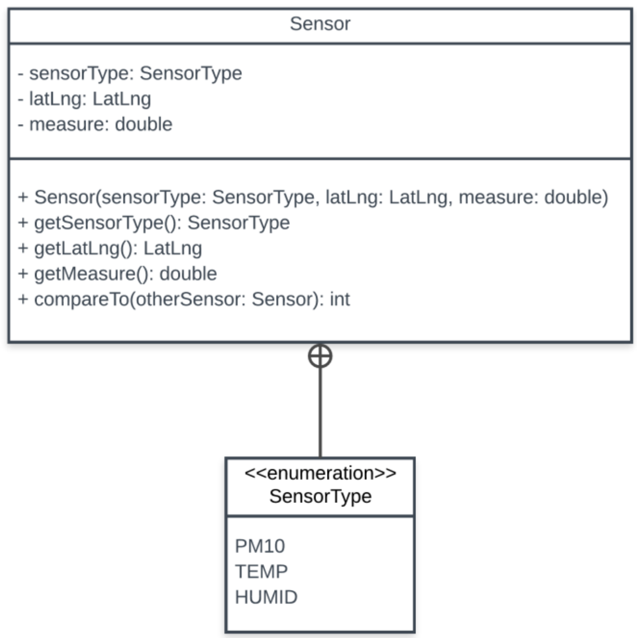


Figure 10. Sensor Class

The sensor class stores generic pollution, temperature or humidity sensors.

Every sensor comprises of a data type, latitude and longitude (for display on Map), and measure.

Comparing between sensors is used to properly order sensors by their measures on Map.

9.5.2.2. History UI

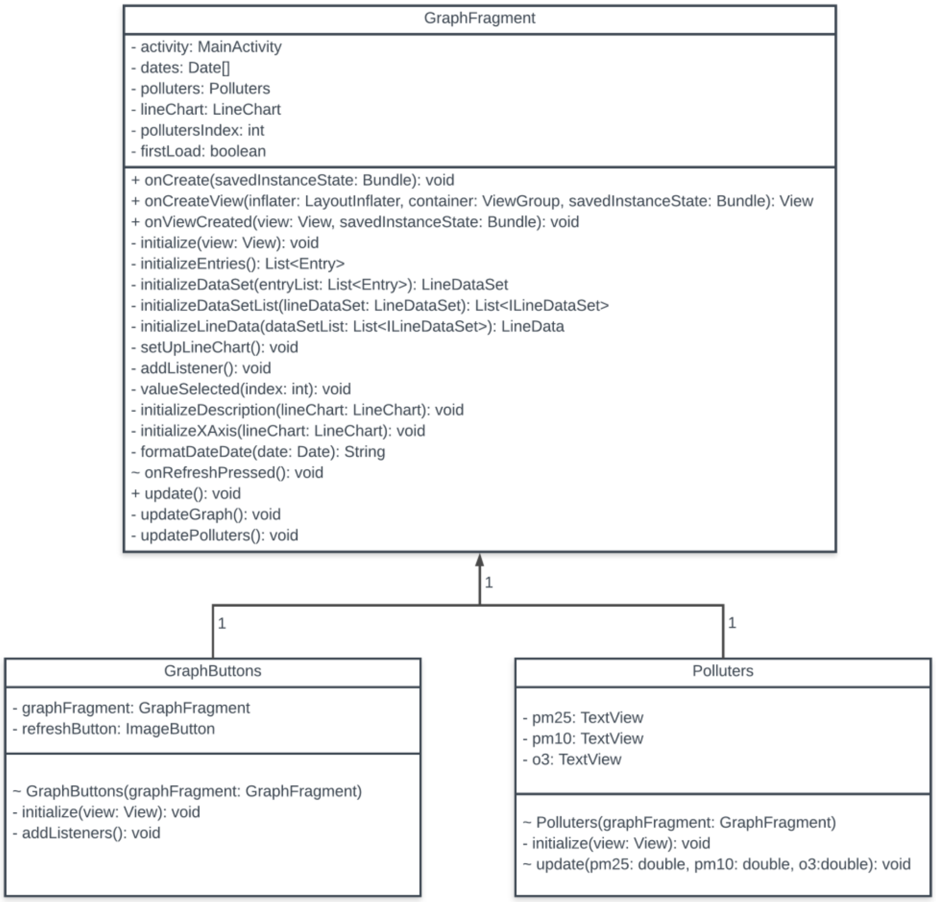


Figure 11. History Fragment component classes

History Fragment (and all other UI interfaces) encourages separation between screen elements.

In this case, the line graph is separated from the screen buttons and detailed polluters (the lower half of the screen).

The Graph Fragment and Polluters classes build the line graph and the overall visual structure, while the Buttons class adds interactivity.

As the public class, GraphFragment is responsible for sending and receiving data update requests.

9.6. Dynamic Behaviour

9.6.1. Sequence Diagrams

As the most complex interaction between data and UI, focus is given on data flow to Home.

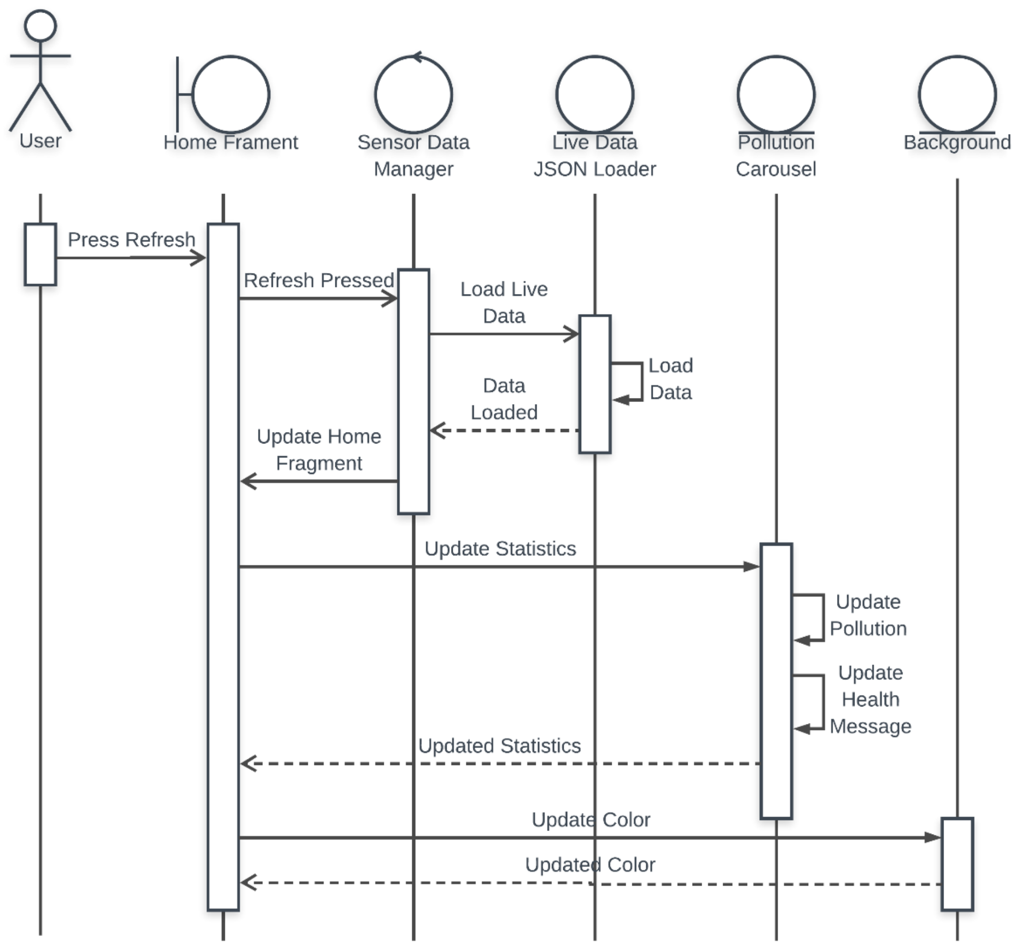


Figure 12. Home Sequence Diagram

Upon pressing Refresh, a signal is sent to the Sensor Data Manager to update all sensor data.
The Manager triggers an asynchronous call to a JSON Loader subcomponent, which establishes a connection to the Urban Observatory database and fetches all available pollution, temperature and humidity sensors. Once data is downloaded and sorted, it is sent to the Home Fragment UI (most notably the carousel).
The displayed statistics update to reflect the new data, and the background colour adapts according to the pollution level.

9.6.2. Activity Diagrams

Illustrated is the execution sequence on program boot-up.

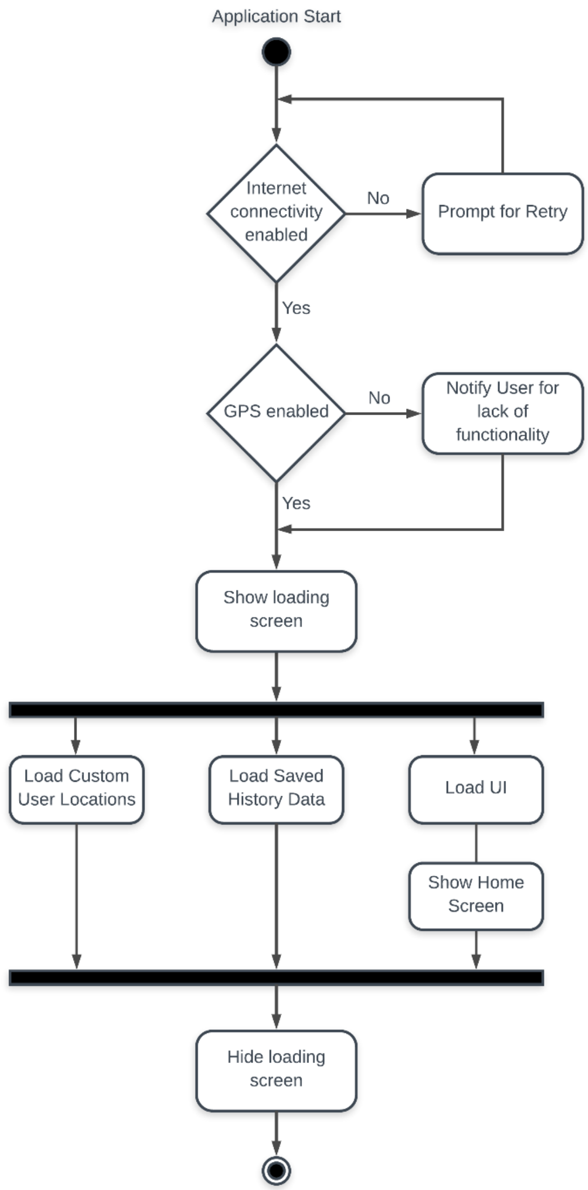


Figure 13. NewAir boot-up sequence

As the program is entirely based on downloading data, it is crucial to perform an Internet connectivity check on start-up. If such does not exists, the sequence is paused and the User is prompted to retry.

An additional feature is gathering data in User proximity, and a GPS check verifies coordinates can be parsed. If connectivity is not available, the User is notified that the feature will not be present, but all other functionalities perform as usual.

NewAir then presents a loading screen, while underlying components load (UI elements are cached, and initial sensor data is downloaded).

9.7. Graphical User Interfaces

9.7.1. Home

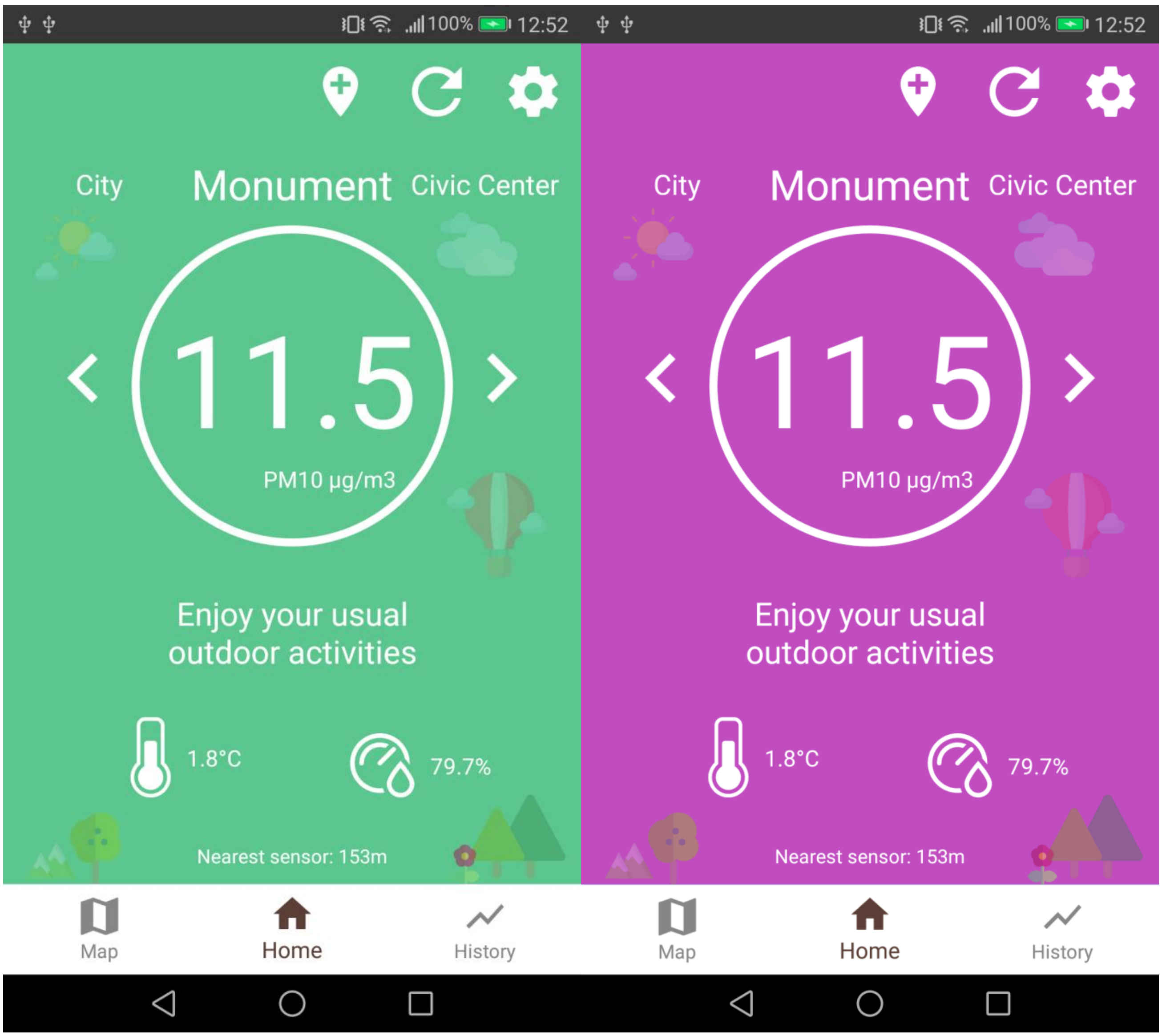


Figure 14. Home screen with default and colour blind mode

Home screen provides immediate access to simplified air pollution, temperature and humidity levels for a set of general and user-defined locations.

Most notable is the background, changing colour according to the pollution level, providing an instant recognition of the air quality without a need of analysing numerical data. In case of colour blindness, a specialised colour blind palette can be enabled in Settings:

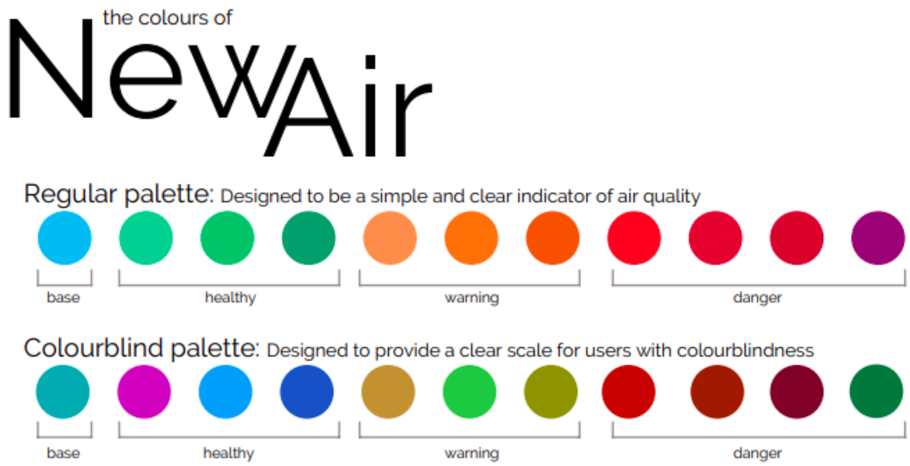


Figure 15. NewAir Colour Palettes

A short text tip advises whether any immediate health-related action needs to be taken with respect to pollution.

The default user location upon start-up is Nearby, using GPS to collect data from sensors nearest to the user. The City location gives the average across all sensors in Newcastle.

Custom user locations can be accessed by either pressing the arrows, or swiping left or right on the locations carousel.

Pressing the map marker button at the top opens a map, where custom locations can be saved. These can later be removed in Settings:

Figure 16. Add Location Screen

Data is updated automatically after a predefined time interval. Modifying the interval is available in Settings. A manual refresh button is also provided.

As this screen is not accessed through the navigational bar, returning to Home is done using the device Back button.

Basic requirements met:

- Offers interactivity by allowing the user to swipe through locations.
- Allows adding new custom locations.

Intermediate requirements met:

- Custom locations are stored on the device, and are loaded upon program start-up.

Advanced requirements met:

- Presents pollution data from the nearest sensor using GPS.

9.7.2. Map

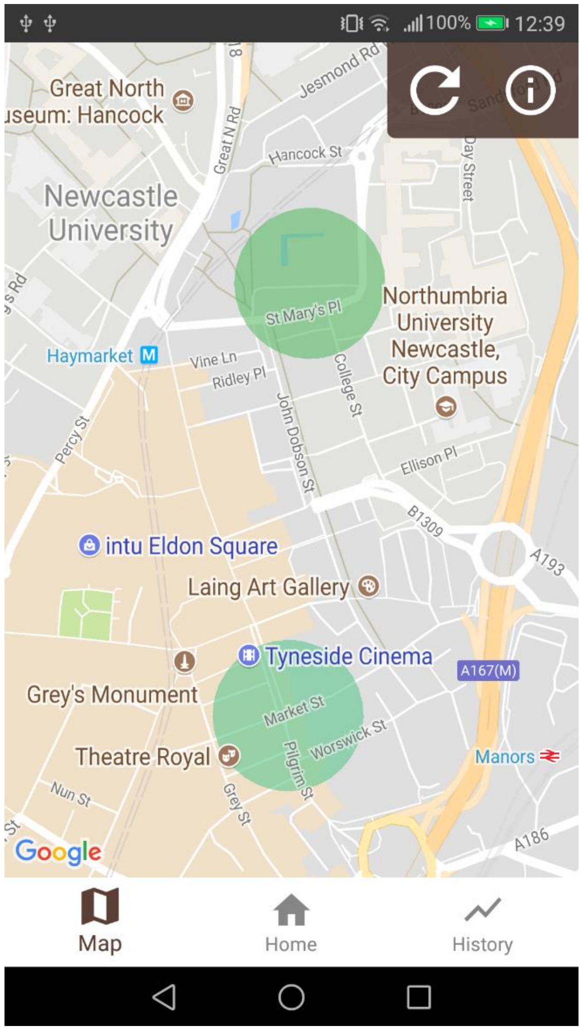


Figure 17. Map screen

Map screen offers an in-depth visual representation of pollution levels across the whole city. The home screen colour palette is overlaid on top of the city map, visualising pollution differences between city regions. Allows for observing air quality for the whole city interactively, abstracting from numerical data.

For example, the above screenshot showcases sensor readings for the same user locations as on the previously shown screenshots.

A legend button at the top reveals a colour legend, allowing recognition of specific pollution levels.

A refresh button offers manual data updates in-between automatic updates.

Basic requirements met:

- Map is interactive.

Advanced requirements met:

- Data is presented visually, rather than numerically.

9.7.3. History

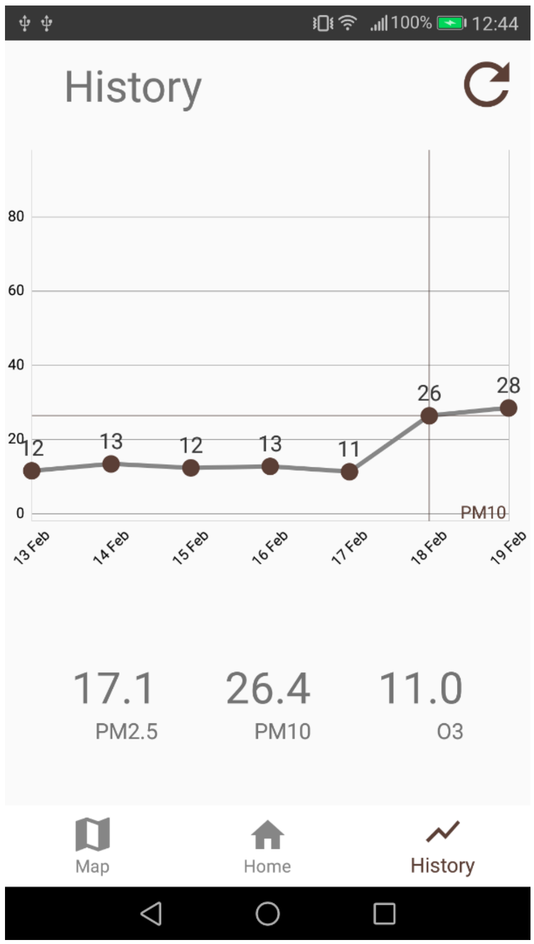


Figure 18. History Screen

History screen offers in-depth numerical pollution statistics for the city over a weeks' time, complemented with a breakdown of pollutants for each timestamp.

Pressing a timestamp updates the polluter data below the graph, pinpointing the pollution levels for the respective day.

Basic requirements met:

- The user can interact with the line graph, choosing from different timestamps.

Intermediate requirements met:

- Data is built over time and accesses previous Urban Observatory records.

9.7.4. Settings

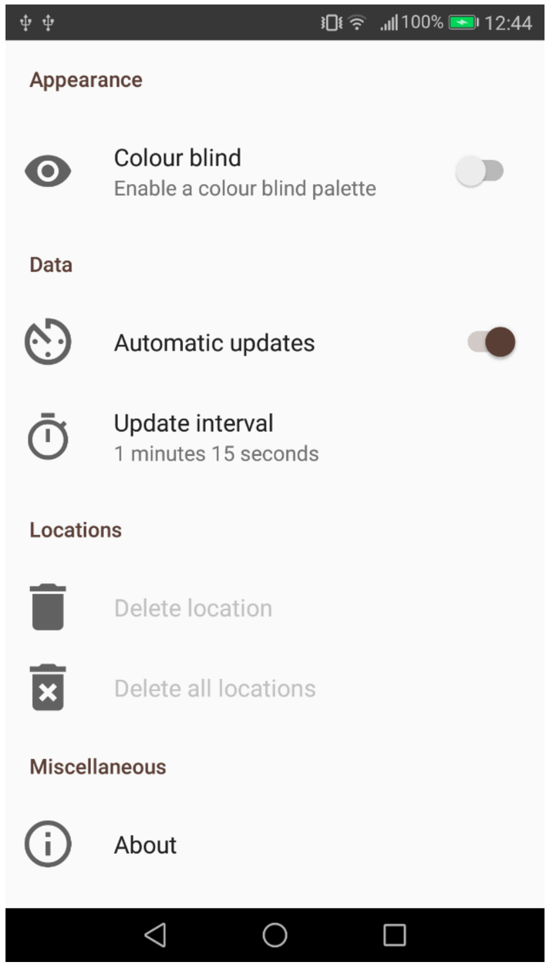


Figure 19. Settings screen

Settings screen allows the user to modify core functionalities and data collection.

The colour palette of the Home screen background can be adapted to colour blind users.

Custom user locations on the Home screen can be removed.

Automatic intervals for fetching sensor data (Timer rate) can be modified by minutes and seconds.

Information about the developers and references to licences and other authors is available.

As this screen is not accessed through the navigational bar, returning to Home is done using the device Back button.

10. Definition of terms

- API - Application Programming Interface - A set of subroutine definitions, protocols, and tools for building application software
- GPS - Global Positioning System – A space-based radio navigation system
- IDE - Integrated Development Environment - A software application that provides comprehensive facilities to computer programmers for software development
- JSON - JavaScript Object Notation - Data-interchange format - An open-standard file format that uses human-readable text to transmit data objects consisting of attribute–value pairs and array data types
- OS – Operating System - System software that manages computer hardware and software resources and provides common services for computer programs
- RESTful API - Representational State Transfer – Web services file format - A way of providing interoperability between computer systems on the Internet
- SDK – Software Development Kit - A set of software development tools that allows the creation of applications for software platforms
- UI – User Interface – The space where interactions between humans and machines occur

Contribution Matrix

	R.J. Cooper	V.M.T. Godsell	I. Gylaris	A.L.J. Heilling	V. Iliev	L.P. Stannard	I.J. Watt	B.D. Wilsher
1. Purpose	C, M , R	C, M , R	C, M , R	C, M , R	C, M , R	C, M , R	C, M , R	C, M , R
2. Background & Analysis	C, M , R	C, M , R	C, M , R	C, M , R	C, M , R	C, M , R	C, M , R	C, M , R
3. Roles and Deliverables	C, M , R	C, M , R	C, M , R	C, M , R	C, M , R	C, M , R	C, M , R	C, M , R
4. Project Plan	C, M , R	C, M , R	C, M , R	C, M , R	C, M , R	C, M , R	C, M , R	C, M , R
5. Hardware and Software Platforms	C, M , R	C, M , R	C, M , R	C, M , R	C, M , R	C, M , R	C, M , R	C, M , R
6. References	C, M , R	C, M , R	C, M , R	C, M , R	C, M , R	C, M , R	C, M , R	C, M , R
7. Solution requirements	C, M , R	C, M , R	C, M , R	C, M , R	C, M , R	C, M , R	C, M , R	C, M , R
8. Other considerations	C, M , R	C, M , R	C, M , R	C, M , R	C, M , R	C, M , R	C, M , R	C, M , R
9. Software Design	C, M , R	C, M , R	C, M , R	C, M , R	C, M , R	C, M , R	C, M , R	C, M , R

C – Create

M – Modify

R – Review

T - Test