Co-funded by the European Union

ESSnet Smart Surveys

Grant Agreement Number: 899365 - 2019-DE-SmartStat

Link to our CROS website

Workpackage 2

Smart Survey Pilots

Deliverable 2.8: : Functional and technical descriptions of tools - Consumption (WP2.1)

Version 1.0, 21-12-2021

Prepared by:

Barry Schouten (CBS, Netherlands)
Sigrid van Hoek (CBS, Nteherlands)
Tom Oerlemans (CBS, Netherlands)
Nick de Wolf (CBS, Netherlands)

Workpackage Leader:

Barry Schouten (CBS, Netherlands)
Jg.schouten@cbs.nl
telephone : +31 70 3374905

SUMMARY: This WP2 deliverable is dedicated to the Household Budget Survey application in WP2.1 Consumption. It provides updated links to the Gitlab repositories for frontend and backend source code. Furthermore, it gives updated descriptions of the CSPA level and the methodology, logistics and legal-ethical levels distinguished in deliverable 2.7. It is stressed that this deliverable overlaps with deliverable 2.9. New and/or revised descriptions relative to deliverable 2.9 are highlighted explicitly. It is also stressed that empirical motivation for specifications in this deliverable are given in deliverable 2.1 a.

1. INTRODUCTION

Deliverable 2.7 proposes four levels of description of smart survey pilots as well as mappings to the WP3 proof-of-concept's and to the six smart survey features. In 2020, functional tests have been performed by six WP2.1 countries (BE, DE, ES, LU, NL and PL) and in 2021 phase 3 field tests have been conducted by three WP2.1 countries (ES, LU and NL). This deliverable is a continuation and revision of deliverable 2.9.

The four deliverable 2.7 levels are:

1. CSPA conceptual, logical and physical levels
2. Methodology level
3. Logistics level
4. Legal-ethical-policy level.

The levels are crossed against frontend and backend. Generic specifications and country-specific features are sketched as well, but provided in detail in deliverable 2.5 on the shareability of tools.

2. DESCRIPTION OF THE HBS APPLICATION

We first note that the HBS application will undergo one final phase of refactoring and cleaning in Q1 2022. This means the descriptions may be changed once more in 2022 as part of Eurostat project @HBS2. We discuss each of the deliverable 2.7 levels.

2.1 CSPA conceptual-logical-physical levels

For the IT frontend and backend, we conform to the inventory following CSPA (Common Statistical Production Architecture) by Eurostat. The inventory can be found at
https://webgate.ec.europa.eu/fpfis/wikis/display/ISTLCS/INVENTORY

NEW: Annex A gives the revised questionnaire for HBS.

The app is listed as Household Budget Survey in Google and iOS app stores and the source code can be found at https://gitlab.com/tabi/projects/budget

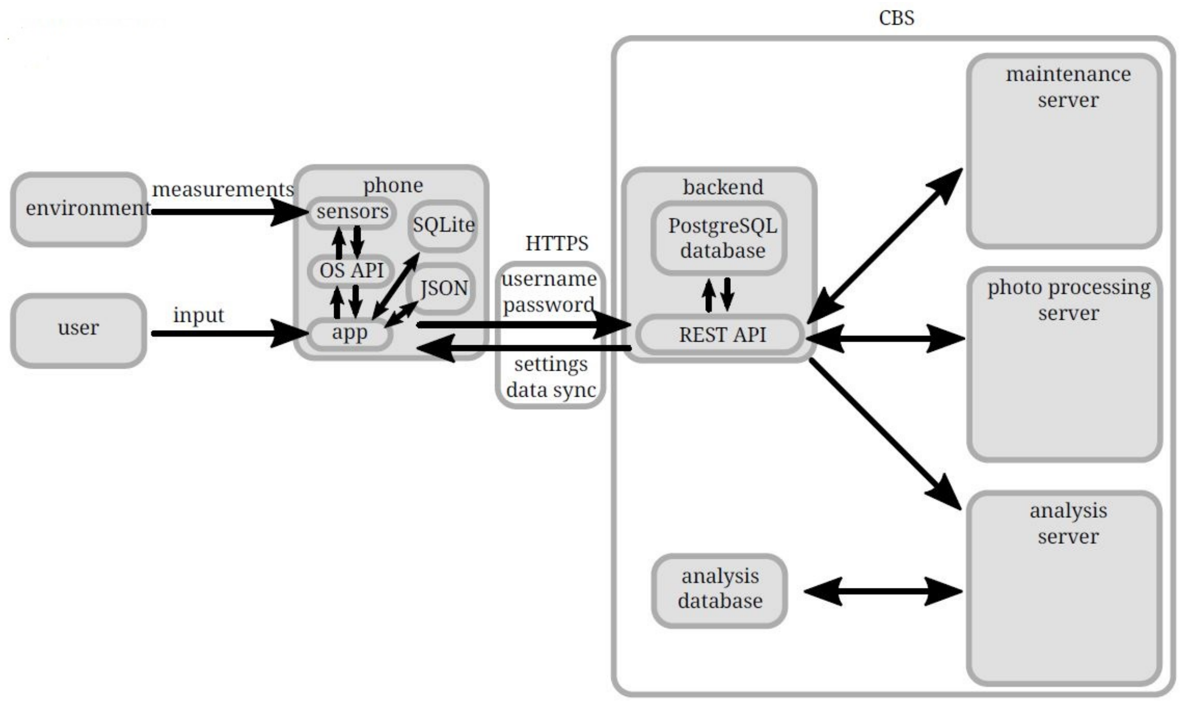
The Household Budget Survey app is at version 5 at the time of writing and two new versions are planned for 2022 and further:

- Version 0: Version prepared to evaluate technical problems across countries and devices (June/July 2019)
- Version 1: Version with revisions of technical issues for test round 1 (August 2019)
- Version 2: Version with revisions based on @HBS test round 1 (December 2019)
- Version 3: Version including receipt scanning pipeline (January 2021)
- Version 4: Version with revisions based on @HBS test round 2 and ESSnet usability tests (May 2021)

- Version 5 (CURRENT VERSION, LABELLED AS 2.1.14 IN APP STORES): Version including introduction questionnaire and in-app OCR feedback and cropping of scan images (Nov 2021)
- Version 6: Refactored version including last issues identified during phase 3 (Apr 2022)
- Version 7 and further: Multi-device data entry (yet to be decided)

The app components and data flows are presented in Figure 1. The app components are the frontend, backend, maintenance server, photo processing server and the analysis server.

Figure 1; Components in the household budget survey application



The components are described one by one.

2.1.1 App frontend

The app is available in 11 country versions: BE, DE, ES, FI, HU, NL, NO, LU, PL, SI, UK. However, the most recent version of the app has only been prepared for ES, FI, HU, LU, NL and SI. Description of the user interface and detailed English screenshots of the app frontend can be found in the former report on the action of project @HBS (Schouten, Bulman, Järvensivu, Plate and Vrabič-Kek 2020). Annex B includes screenshots of the main screens.

Generic features of the app frontend are:

- Diary functionality
- Manual data entry and receipt scan data entry
- The use of separate product search lists linked to COICOP
- The use of separate store lists to prepare receipt processing and classification of products
- Four main screens: calendar, list of submitted expenditures, expenditure statistics and settings
- Check of receipt scan data quality
- Feedback of text extraction of scanned receipt

- NEW: In-app image cropping of receipt scans and respondent interaction
- NEW: In-app tutorial movies and screens
- NEW: In-app reminder/notification option

Implemented configurable features of the app frontend are:

- Length of data collection period
- App language
- Content of product search lists
- Content of store search lists (names and types)
- Helpdesk functionality (helpdesk phone number and email)
- Timing of statistics feedback to respondent (instantaneous, delayed, not)
- NEW: In-app OCR feedback and scan image cropping
- NEW: Logging of in-app paradata on navigation behavior
- NEW: Introduction questionnaire
- NEW: NSI logo and country-specific app store name
- NEW: In-app country-specific OCR/NLP

Not yet implemented configurable features are:

- App colour scheme

Features that some countries prefer to be included (will be detailed in deliverable 2.5):

- Quantities and number of units of purchased products
- Within household filtering of expenditures
- Within household anonymization
- Distinction of biological/ecological products
- Inclusion of homegrown products

2.1.2 App backend

A preliminary description of the HBS backend was provided in deliverable 2.9. Since then, the backend has been implemented and expanded, in particular for in-app paradata and for handling of processed receipts. The backend communicates via a REST API interface with the apps on the respondent's phones as well as the maintenance-, photo-processing- and analysis-servers. Communications between the REST API and the apps is encrypted via HTTPS and authenticated with usernames and passwords.

See Annex C for a detailed description of the backend.

NEW: A description of the backend database structure can be found at
https://gitlab.com/tabi/projects/budget/-/blob/master/documents/backend.md

2.1.3 Maintenance server

The maintenance server has three crucial tasks:

- Case management:
  - Assignment of app protocols

    - Generation of users and passwords
    - Initiation of contact, reminder and motivation strategies
- Monitoring of data collection:
    - Linkage to sampling frame and administrative data
    - Queries on recruitment and participation rates
    - Queries on in-app answer and navigation behaviour
- Helpdesk support:
    - Background on technical errors or other data entry issues
    - Checks of conditional incentive requirements

The maintenance server is accessed through a dashboard web application. Access to the various tasks is dependent on roles and permissions assigned to a person.

Figure 2: Use-case diagram of maintenance server

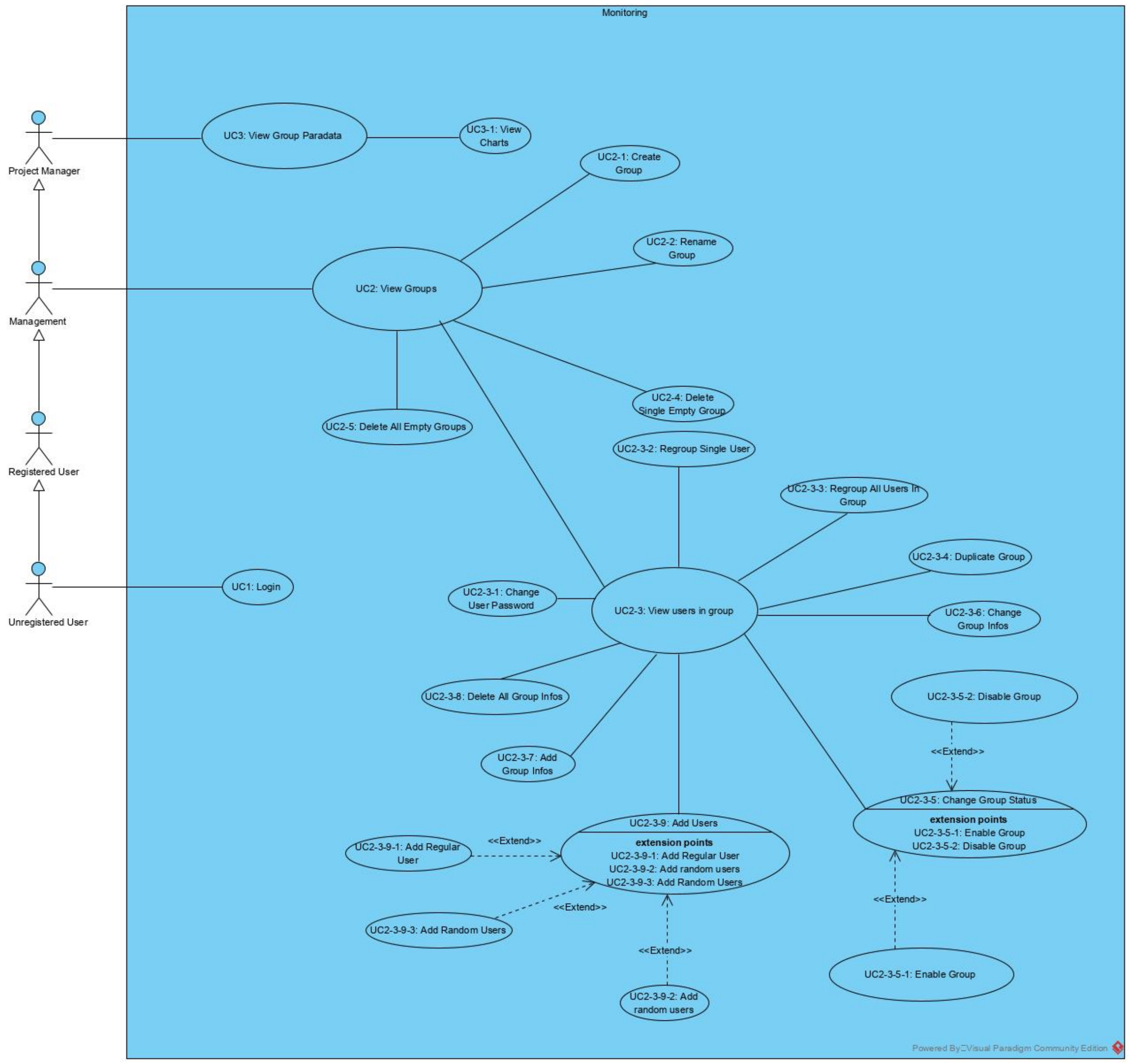
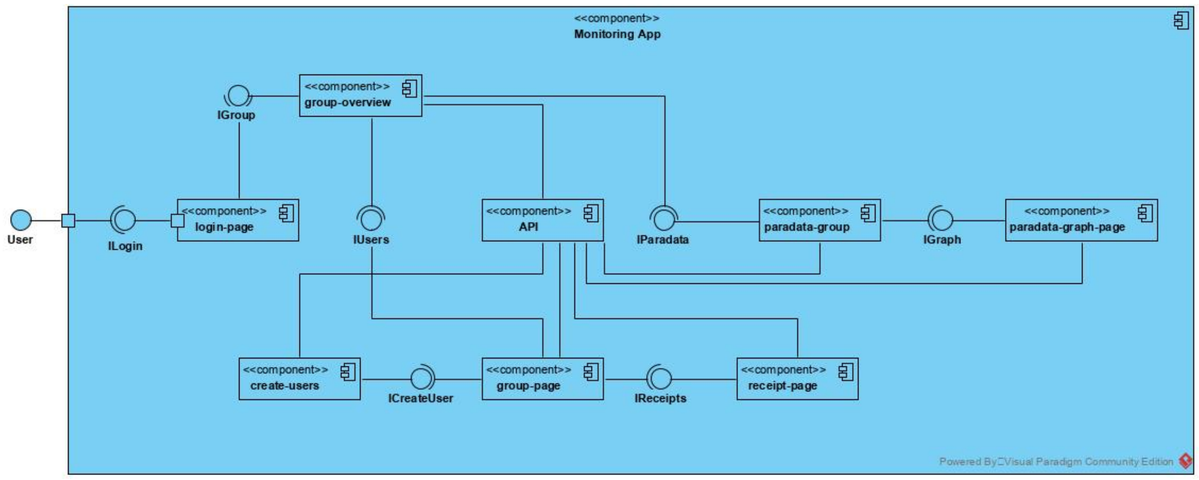

Monitoring
Project Manager
Management
Registered User
Unregistered User
UC3: View Group Paradata
UC3-1: View Charts
UC2-1: Create Group
UC2-2: Rename Group
UC2: View Groups
UC2-4: Delete Single Empty Group
UC2-5: Delete All Empty Groups
UC2-3-2: Regroup Single User
UC2-3-3: Regroup All Users In Group
UC2-3-4: Duplicate Group
UC1: Login
UC2-3-1: Change User Password
UC2-3: View users in group
UC2-3-6: Change Group Infos
UC2-3-8: Delete All Group Infos
UC2-3-5-2: Disable Group
<<Extend>>
UC2-3-7: Add Group Infos
UC2-3-5: Change Group Status
extension points
UC2-3-5-1: Enable Group
UC2-3-5-2: Disable Group
UC2-3-9: Add Users
extension points
UC2-3-9-1: Add Regular User
UC2-3-9-2: Add random users
UC2-3-9-3: Add Random Users
UC2-3-9-1: Add Regular User
<<Extend>>
<<Extend>>
UC2-3-9-3: Add Random Users
<<Extend>>
<<Extend>>
UC2-3-5-1: Enable Group
UC2-3-9-2: Add random users
Powered By Visual Paradigm Community Edition

Figure 3: Component diagram of maintenance server



The maintenance server is implemented in Angular. The use-case diagram in Figure 2 and the component model diagram in Figure 3 show the main functionalities of the maintenance server:

- Login
- Management of survey groups
- Management of users in survey groups
- Monitoring survey progress on group level (paradata)
- Monitoring survey progress on user level (receipt)
- Monitoring usage of HBS app (paradata)

NEW: Annex D gives a guide to the web application. Furthermore, source code can be found at the HBS git repository https://gitlab.com/tabi/projects/budget

2.1.4 Photo processing server

The photo processing server handles the processing of receipts scanned by respondents. Receipt scanning has the following steps:

1. In-app scanning
    a. Taking a picture
    b. Check of scan quality to interact with respondent
    c. In-app OCR to interact with respondent
2. In-house OCR and language processing
3. In-house classification to COICOP
4. In-app feedback of classified receipt

The in-house codebase can be found in the following gitlab repository:
https://gitlab.com/Nickdewolf/hbs.wp4-shopping-receipt-recognition

The in-house codebase uses Tensorflow and OpenCV for some of the image pre-processing and Tesseract for the OCR. For COICOP classification we used a library called FastText. All pre-trained models are included in the gitlab repository.
The code has been divided into four categories: *mob*, *ocr*, *coicop*, and *api*.

1. ***mob***: Handles the image processing steps, such as background removal, skew correction, and brightness/contrast/color improving.

2. ***ocr***: Handles the steps involved with OCR and language processing. The functions in these files call Tesseract on the pre-processed images and return raw text, after which this is further processed to retrieve informations such as purchased products and their corresponding prices.
3. ***coicop***: Handles the training classifiers and contains the functions to classify product descriptions into their corresponding COICOP code. The current pre-trained model can handle 112 different COICOP codes.
4. ***api***: Contains all the code for retrieving and sending data between the backend. This files uses the data as contained in requests_config.json to find the correct urls.

Some example code has been included in the form of jupyter notebooks:

- **00_Prepare_CBS_data:** shows example code on how we processed the internal data, which for privacy reasons cannot be shared on the gitlab repository.
- **01_Train_Models:** shows example code on how to train various models for the COICOP classification. The input .csv files requires just 2 columns: the first containing product descriptions and the second containing the corresponding COICOP code.
- **02_Test_Pipeline:** Shows code for each of the steps included in the pipeline, such as: retrieving an image from the server, apply image processing, apply OCR and language processing, COICOP classification and preparing data to be sent back to the backend database through the rest API.

To run the code with the previously mentioned backend, the **main.py** can be used to query the server for changes every 5 minutes. This file essentially runs code similar to the code found in 02_Test_Pipeline.ipynb with additional error handling.

2.1.5 Analysis server

The analysis server is the link to the statistical analysis. It is the starting point for further data processing and production of statistics. It contains no intermediate survey and sensor data and only data of completed data collections.
The Analysis server and database are not implemented yet. While this functionality is not available the input database can be queried directly with the pgAdmin_4 tool for PostgreSQL databases.

NEW: Statistical queries employing both survey data, receipt scan data and in-app paradata have been prepared in R and will be posted on the git repository.

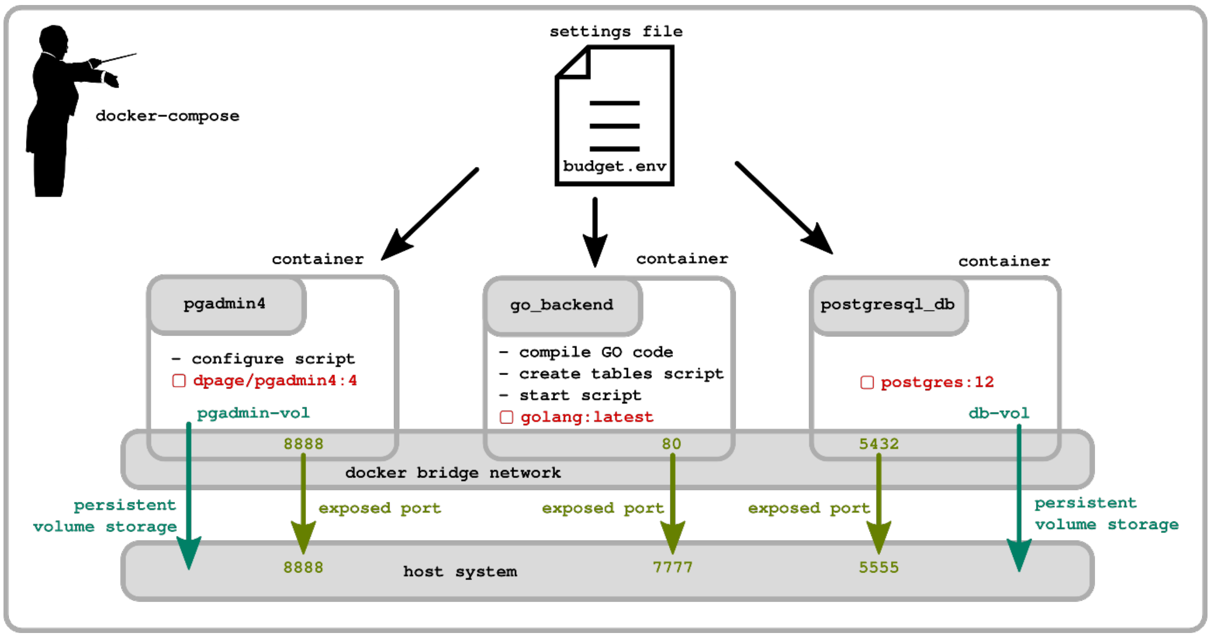
2.2 Backend deployment

The backend can be deployed in micro-services. Every logical component is put into a Docker container. These containers are then deployed – and tied together – in a set. A single settings file contains the configuration options for all containers. The orchestration happens via docker-compose.

The docker-compose.yml file contains all the glue that holds the containers together and makes them operate as a team. It takes care of building the containers (if necessary), setting up an isolated network for inter-container communication, exposing container ports to the host system (so that one can still interact with the micro-services individually) as well as creating persistent storage that remains intact even when the ephemeral containers are destroyed.

So far the orchestration includes a PostgreSQL database, the database administration tool PGadmin4 as well as the GOlang REST API. It can easily expanded to include the maintenance server or the photo processing server as well. See Figure 4.

Figure 4; Deployment of the backend components with Docker and Docker-compose



2.3 Methodology level

The methodology level consists of:

- User interface of frontend: The most influential design choices in the user interface, including 'gamification' features.
- Data collection strategy: This concerns the use of contact modes, contact and reminder strategies, incentive strategies, recruitment materials, the use of "non-smart" modes.
- Data quality checks: Soft and hard checks of plausibility of entered data and notifications of missing data.
- Machine learning models in processing or mixing/fusing sensor data, including pre-training of models and re-training of models through active learning (with the help of human annotation and validation procedures) or online learning (automated).

We discuss each of the levels for the household survey application. Detailed elaborations and empirical motivations for some of the choices and recommendations will be provided in deliverable 2.1 a.

User interface influential choices:

- Search algorithm: Manual data entry of products can range from fully open to fully closed. Under a fully open data entry, respondents type product names and these are classified later. Under a fully closed data entry, respondents can only choose from a list. There are various hybrid options. The way the matches are determined and displayed can be configured in the HBS app.

NEW: The HBS lets respondents type products and then shows matches based on the Jarro-Winkler distance. At most ten matches are shown and at most two per COICOP, and matches are only shown if the distance is smaller than or equal to three.

- Product lists: In a closed or semi-open manual data entry, product lists language and diversity determine the ease with which respondents can find products. It can be decided to include frequently occurring typos, even though they are grammatically incorrect. The quality and richness of lists varies greatly between countries.
  NEW: The HBS app contains updated product lists for ES, FI, HU, LU, NL and SI.
- Shop lists: To aid receipt processing, a list of shops can be added. Shops have different receipt formats. Shop lists are not common in ESS countries.
  NEW: Shop lists have been added for ES, HU and LU.
- Discount/rebate: Products that have a discount or are rebated complicate data entry both manually and through scans. Choices how to present discount and rebate turned out influential in usability testing.
  NEW: The implementations for discounts and rebates have been redesigned according to suggestions made in usability tests. For receipt scans, the recommended option is to let respondents check. NB: Detailed motivation is provided in deliverable 2.1 a.
- Receipt scan data entry (NEW): The HBS app allows for scanning and submission of receipts. In 2021, this part of the UI has gone through five major changes: 1) the app gives hints to the respondent on light and contrast conditions while in camera mode, 2) the app allows for an automated receipt detection while in camera mode, 3) the receipt scan image can be cropped, 4) in-app OCR and language processing results can be displayed and edited by respondents, and 5) an in-app OCR confidence score is computed that can be compared to a configurable threshold that prompts a message. The preliminary conclusion is that changes 1, 2, 3 and 5 greatly improve usability, but that change 4 needs further analyses to reach an optimal trade-off. NB: Detailed motivation will be provided in deliverable 2.1 a and deliverable 3.3 of ESTAT project @HBS2.

Data collection strategies:

- Interviewer assistance: Interviewers can be involved in recruiting respondent and keeping them motivated. Interviewers may be involved right from the start or only after nonresponse. In the ESSnet two strategies have been tried; one without interviewers and one where interviewers visit household to recruit them.
  NEW: Conclusion is that interviewer-assisted recruitment and motivation are the preferred strategy. Recruitment and completion rates are 10% higher when interviewers drop-off material. NB: Detailed motivation will be provided in deliverable 2.1 a.
- Recruitment material: Since app data collection requires downloading and registering an unknown app, recruitment material includes instructions, basic overview of screens, possibly a landing page and possibly a brochure explaining what data is collected and for what purpose. For the HBS recruitment without interviewers, recruitment material consists of a manual with overview of screens and a brochure displaying what kind of statistics respondents will get about their own expenditures. In-app per screen there is a brief tutorial on how to navigate.
  NEW: Recruitment material has not been altered. It is concluded that respondents can best be pointed to the app directly rather than through a landing page. An in-app tutorial movie was added that is automatically launched when first scanning a receipt. The movie gives instructions how to select products and prices on receipts, which is especially helpful for long receipts.

- Multi-mode HBS: HBS can be a mix on app data collection, online data collection through a website, interviewer questionnaires and paper diaries. Modes/devices may be offered concurrently or sequentially. In the HBS field test, only the app itself will be used, but accessible on smartphones, tablets and laptops/desktops. In other field tests, more advanced multi-mode strategies are tested.
  NEW: Although not tested through experimental conditions it seems advisable that alternative modes are offered such as paper diaries or submission of receipts by postal mail. The choice of modes depends strongly on the country. The HBS app may be used to scan, process and store receipts in-house.
- Household reference person: HBS is a household survey. On one extreme one household member completes all questionnaires and diary and, thus, also provides info on expenditures by household members. On the other extreme, all members submit expenditures individually. An important design decision is whether household members can view each other's expenditures. In the current HBS app, all household members can submit expenditures but there is no anonymization. The data collection strategy will be to assign one contact or reference person who does most of the data entry and makes sure other members submit as well.
  NEW: Preliminary analyses suggest that most households participate through a single device. NB: Detailed empirical results are given in deliverable 2.1 a
- Incentives: HBS is a burdensome survey and many countries include unconditional and/or conditional incentives. There is, however, great variety in the amount of incentive. In the HBS field test this is mostly left to countries. As a general incentive the respondents get in-app feedback. This feedback consists of two graphics, a circle chart and a time series chart. The feedback can be instant, delayed to the end of data collection or omitted completely.
  NEW: Analyses seem to suggest that offering insights does not increase response or completion rates. Impact on data quality still is under investigation. NB: Detailed empirical results are given in deliverable 2.1 a.
- Reminders (NEW): Respondents can set reminders through the settings screen of the app. They are not explicitly pointed at this option unless they start the tutorial under the settings screen. However, the notification option features prominently on the settings screen. Once enabled, the app sends a notification each day at a specified time. NB: In deliverable 3.3 of ESTAT project @HBS2 detailed analyses of in app navigation behaviour are presented.

Data quality checks:

- In-app scan quality: Scans of receipts may vary greatly in quality depending on light conditions, camera settings and type and respondent behaviour. Currently, the HBS app informs the respondent on scan quality while taking a picture. It also displays a bounding box that the respondent can adjust. Furthermore, it provides in-app OCR so that the respondent can decide whether the scan is good enough or needs to be retaken.
  NEW: Analyses suggest that all quality checks while in camera mode are helpful to respondents. However, it still has to be investigated whether offering edits of text extraction results is beneficial. In-app confidence scores for OCR are helpful in post-survey analyses. NB: In deliverable 3.3 of ESTAT project @HBS2 detailed analyses of in-app navigation behaviour and consequences for data quality are presented.
- Checks of amounts and process: Respondents may produce errors in number of products and or prices when entering data manually. Currently, there is no check on outliers, but

the total amount is calculated and shown to the respondent. This can be seen as a soft check. Also in the individual statistics, outliers may be noticed.

- Plausibility of expenditures: Expenditures of households are unknown and may take any form from diverse and many to monotone and few. It is unlikely that a respondent will not buy food for many weeks, but not impossible. The current HBS, therefore, only asks respondents to validate each day. It asks whether all expenditures are included. Respondents can ignore these reminders and validate a day without expenses.
- Respondent validation: Respondents may get classified receipts from the backend. When being online, the feedback is available within 10 seconds. If the respondent works offline, then processing is done at the first instance the respondent goes online. The classified receipts may be edited, but it is still an open question what should be editable and what not.
  NEW: While preparing the field test it was decided not to include feedback on classification. Quality of the results was unclear, especially across countries and respondents may not know what to do if a classification is wrong. Feedback of classification remains a topic of further testing. In-house OCR, language processing and classification do provide confidence scores that can be employed to initiate manual checks.

ML models:

- OCR and Natural Language Processing: See earlier descriptions. The HBS applies in-app OCR as a form of pre-scan of data quality and in-house OCR for processing. In time the in-house OCR may be included in the app itself. The ML model which is sizeable can be compressed.
  NEW: A preliminary conclusion is that both text extraction and classification should be done in-house. The in-app text extraction is especially useful for quality checks. NB: In deliverable 3.3 of ESTAT project @HBS2 detailed analyses of receipt scan processing and consequences for data quality are presented.
- Classification: See earlier descriptions. Classification of products can be in-house and in-app as well depending on how the ML models are trained. There are three options. The first is to train ML models on annotated receipts. This could be done through active and online learning, making sure that models are re-trained. The second is to use EAN/GTIN product descriptions as typically available in scanner data and to train models to classify to these. The descriptions are linked to COICOP by CPI departments on a regular basis. The third is where receipt texts are directly obtained from shops. The last option is ideal but leans heavily on cooperation of shops. The second option is used for NL and some other countries. Countries without scanner data or limited scanner data have to resort to the first option. In the second and third option, ML models have to be applied in-house as shops demand that these data are not exported.
  NEW: Another classification option emerged during 2021, namely the use of texts printed on receipts (including link to EAN and COICOP). This option is now available at Stat NL and facilitates string matching instead of less accurate machine learning approaches. The use of receipts texts provided by shops is still under investigation but is very promising. It is estimated that per year roughly 10 to 20% of texts are new. Also there are seasonal influences during holidays. NB: In deliverable 3.3 of ESTAT project @HBS2 detailed analyses of various receipt scan processing options are discussed.

2.4 Logistics level

The logistics level distinguishes:

- Recruitment: Many of the ESS surveys have an interviewers-assisted data collection such as doing the starting questionnaire, recruiting and assisting respondents in a diary, motivating respondents during data collection, and/or picking up closing questionnaires.
- Monitoring: Sample units may not participate, drop-out or deliver insufficient data quality. Monitoring dashboards may be inspected on a frequent basis in order to determine whether interventions are needed at overall level.
- Assistance: Respondents may be assisted in starting and using smart survey tools passively through a helpdesk and website or actively through interviewers and or technical experts.
- Human-in the-loop machine learning: In sensor data applications, models seldom reach 100% accuracy. Certain population subgroups or certain survey statistics may require manual inspection. In ESSnet Smart Surveys where feedback of statistics to respondents is deemed important, such human-in-the-loop processes may even occur during data collection.

We give a first exposition of how the different logistics are implemented for the household budget survey application.

Recruitment:

- Interviewer involvement training: Most countries employ interviewers to recruit respondents. This interviewer role, implies that interviewers are themselves experienced app users and should be able to answer at least the most basic FAQ. It also means that they should periodically be instructed when the HBS app changes.
  NEW: Interviewer assistance turned out valuable in terms of recruitment and completion rates. During fieldwork, interviewers do give assistance to respondents having trouble to log in or use certain app features. This implies that training materials include FAQ and information on how to use the app. Interviewers should get sufficient time to get acquainted with the app. It is recommended that they work with the app for a week themselves. NB: Detailed results on interviewer assistance during data collection are discussed in deliverable 2.1 a.
- Interviewer guidance: Some countries let interviewers keep track of respondents during data collection. The role of monitoring respondents implies that app data collection monitoring is accessible by interviewers through frequent reports or through direct access to the backoffice. It also implies they must be even more knowledgeable about how the app works and what issues may occur. Currently, in the HBS there is no access of interviewers to the backoffice. For now it is planned that they get weekly updates.
  NEW: Interviewer assistance turned out valuable in terms of motivation of respondents during fieldwork. Interviewers need to be updated on progress so that frequent status reports to interviewer staff are needed. During phase 3 field tests, three status reports (Monday, Wednesday, Friday) have been prepared.

Monitoring:

- Recruitment and activity: Monitoring of registration and submission of data are fairly standard. It means that on a continuous basis queries can be run. In the maintenance server this is indeed possible.

NEW: For HBS a monitoring dashboard has been developed (see Annex D) that allows for queries on in-app activity. Weekly monitoring is recommended at the early stages of data collection.

- Answer and navigation behaviour: Monitoring of in-app behaviour is not at all standard. This concerns consultation of certain app screens and help options, time lags when filling in expenditures, problems when scanning receipts, and so on. Currently, the HBS app has a paradata plug-in that logs all behaviour, but this needs to be made more sophisticated and targeted at certain behaviour.
  NEW: For HBS a monitoring dashboard has been developed (see Annex D) that also allows for queries on in-app navigation behaviour. Detailed paradata can be logged providing overviews of screen times, all in-app clicks, warning messages and use of manual and scan data entry. Frequent monitoring is recommended at the early stages of data collection.
- Technical errors: Respondents may experience problems that are specific to their own devices or to general system failure such as the backend server being inaccessible. These errors can partially be logged. Currently, type of device and operating system are derived and can be used in detecting problems.
  NEW: In part, in-app paradata allows for evaluation of potential technical errors as toast messages prompting warning messages are included in paradata logging. Furthermore, Apple and Google allow for analytics and error evaluations.

Assistance:

- Survey helpdesk: Respondents may call the NSI (as usual) to ask questions about the purpose of the survey and about the task that they are supposed to (such as definitions etc). This type of assistance is not very different from the traditional HBS, except for user interface questions.
  NEW: The HBS app allows for a helpdesk telephone call or email message through the settings page. A click on the option is sufficient to start a helpdesk call. In the phase 3 field tests, questions to the telephone helpdesk have been modest in numbers. NB: Detailed results on helpdesk consultation during data collection is discussed in deliverable 2.1 a
- Technical helpdesk: When respondents experience problems, then these are generally too complicated for the regular helpdesk. Currently, the respondent can send messages in-app to the NSI. Obviously, this only is available when the app is installed. The technical helpdesk email is also available in recruitment letters but cannot be contacted by phone. The survey helpdesk can forward questions.
  NEW: The HBS app allows for an email message through the settings page. The email message contains device type and OS so that potential issues can be evaluated. In the phase 3 field tests, questions to the technical helpdesk have been modest in numbers. NB: Detailed results on helpdesk consultation during data collection is discussed in deliverable 2.1 a
- Landing page/FAQ: It is recommendable to have a dedicated webpage for the survey including explanations, instructions and FAQ. In previous experiments, it was experienced that respondents do not make much use of the webpage and tend to use in-app options or contact the helpdesk.

Human-in-the-loop ML:

- Inspection of classified receipts: It is yet an open decision to what extent OCR and classification of receipts contains a human check. Both the OCR and classification return

indicators of accuracy. Based on lower thresholds to these indicators, it can be decided to flag processed receipts for inspection. This may be country-dependent as ML models for classification depend heavily on the quality of training and retraining of models. For now, it is foreseen that a manual check of performance is included until more experiences are available.
NEW: Based on preparations and preliminary analyses it seems recommendable to perform learning and retraining only in between waves of the survey. Manual data entry of products that are not recognized can be annotated by respondents. Products unknown to ML classification may be gathered to inform manual checks and active learning. Online learning is warranted to account for the dynamics in products over time. NB: Detailed discussion on receipt processing options is provided in deliverable 3.3 of ESTAT project @HBS2.

2.5 Legal-ethical-policy level (preliminary)

The legal-ethical-policy level is in part still under discussion. It is the topic of the ESSnet working group legal-ethical. This working group plans to submit a request to the EDPB (European Data Protection Board) on smart surveys. This concerns broader questions like quality control and quality metadata and trade-offs between in-device versus in-house processing and storing of data. These discussions especially affect the receipt scanning pipeline of HBS.

Statistics Netherlands has performed a risk assessment as input to a dedicated data protection impact assessment (DPIA). The risk assessment is available in English. As a last step a penetration test of the HBS backend has been prepared by an external party. The report of this pen test will become available in Q1 of 2022. The DPIA itself is already available.

A number of relevant recommendations can be made:

- Image cropping: Scan image cropping is recommended to enable respondents to select the parts of receipts that contain products and prices. This can, however, not be enforced, so that potentially still irrelevant but partly identifying information is scanned and stored. An in-app tutorial movie is shown to respondents when scanning for the first time that instructs them to select only relevant parts of receipts.
- Threshold on OCR confidence score: In order to avoid low quality scans or scans of other objects than receipts, a configurable threshold should be set on in-app text extraction. Even if in-app processing of receipts is not used to derive products-prices themselves, such a threshold is a first security check.
- Editing of receipts: It is recommended that respondents can delete and replace scans. It is yet an open decision whether respondents should also be able to edit text extraction from receipts.
- Digital receipts: In order to not exclude or disadvantage respondents that have digital receipts, it is necessary that such receipts can be submitted. The current solution in the HBS app is to scan such digital receipts through a second device.
- Device types: The current practice is that devices up to five years old are supported. During the field test it turned out that this period should be made longer.
- Submission of receipts through alternative channels: In some countries there is a clear distrust against government apps, in part due to COVID-19 developments, leading to relatively low recruitment rates for an HBS app. It is recommended that in such settings, respondent should

remain to be able to submit data by alternative channels such as postal mail or a website. The HBS app can be used to process paper receipts in-house.

Open decisions:

- In-app editing: It has yet to be determined whether the data quality of in-device text extraction is sufficiently strong to perform in-app processing of receipts. In phase 3 field tests, the option to edit has been randomized across sample units and is under evaluation.
- In-app versus in-house processing: In-app classification of receipts texts does not seem to be a feasible approach for two reasons. The first reason is that CPI/transaction data is needed that are not allowed to leave NSI premises. The second reason is that the app would increase strongly in size. The only viable choice seems to be between in-app extraction of products and prices and in-house text extraction. However, this opens up the debate of data quality and informative quality metadata.

3. RELEVANT INFORMATION FOR WP3 PROOF-OF-CONCEPT AREAS (NEW)

Per PoC area relevant conclusions are drawn:

POC on architecture-metadata

- The current app-assisted approach differs from a regular online HBS in two major ways: 1)The handling and processing of receipt scans/digital receipts is added as a smart feature, and 2) storage and processing is partly done locally. The use of private/personal devices for HBS is in principle similar to other app-assisted smart surveys when it comes to security and architecture. The receipt handling and processing is, however, specific to HBS and requires two main services: text extraction (OCR and language processing) ad product classification.
- The current app-assisted approach differs in one more fundamental way from a paper HBS: Product search algorithm/matching is added as a smart feature and assists respondents and the NSI in classification to COICOP.
- In the learning stages in which the HBS still is and will remain for the near future (given extensions to digital receipts and data donation) in-app logging of navigation behavior provides powerful information on potential ways to improve data quality. The current HBS thus assumes paradata logging which can be made configurable from basic screen time info to all in-app clicks/actions. Paradata have their metadata which may be regarded as quality metadata.
- The HBS app assumes a fully decentralized back-office. Each country creates its own version of the app that points to a NSI URL. What should be shared are UI design and all kinds of services linked to product search, receipt processing and questionnaires. These services are likely to be broadened when uploading of digital receipts and data donation are added.
- Receipt scan quality metadata are crucial to HBS. It is recommended to store confidence scores for products and prices and for classification to COICOP. Currently, also scores for in-app text extraction are transmitted to the HBS backend.

POC on privacy preservation

- In-device/app classification of receipts does not seem to be feasible since data are involved that cannot leave NSI premises
- PET/PPT techniques do not seem to be especially suited for HBS receipt processing as extracted products and prices are merely half-products that still require classification. Given

that typos and other small errors that occur during text extraction may be overcome in classification, any aggregation of information is disadvantageous. Only when in-device extraction can be made very accurate it is that PET/PPT might be applicable.

POC on machine learning

- It seems best to apply pre-trained machine learning approaches to receipt text extraction and product classification during fieldwork.
- Online learning approaches need to be applied to account for dynamics in products over time. These should be applied in between rounds of HBS and are country specific.
- Active learning approaches need to be applied when unknown products are submitted during the HBS data collection. This can be done by respondents in manual data entry by allowing them to add new products. These products then need to be linked to formal classifications by HBS staff post-survey. For receipt scan data entry products with classification probabilities below a specified threshold can be set aside for manual inspection.
- Federated learning of classification would thus take place at the country level and in between waves of HBS.

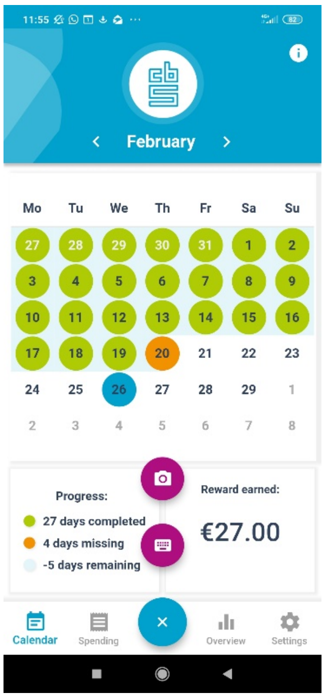
POC on gamification

- ‘Gamification’ features in the current HBS app are monetary incentives, display of progress, insights on expenditures, in-app options to edit OCR results and user interface design. These still need re-evaluation from a gamification perspective in terms of what respondent tasks are most complicated and perceived as most burdensome. It should be evaluated how the respondent role in active learning through editing receipts and labelling unknown products can be made as enjoyable as possible. Furthermore, it should be evaluated how scanning of receipts can be made more attractive.
- It has yet to be determined whether offering personalized insights on expenditures is beneficial on data quality. First results suggest it is not strongly influential on recruitment rates.
- If gamification principles are applied then this must be combined with interviewer training and design of recruitment materials.

ANNEX A – EUROSTAT CSPA INVENTORY QUESTIONNAIRE
See attachment

ANNEX B – SCREENSHOTS OF MAIN APP SCREENS

Calendar:

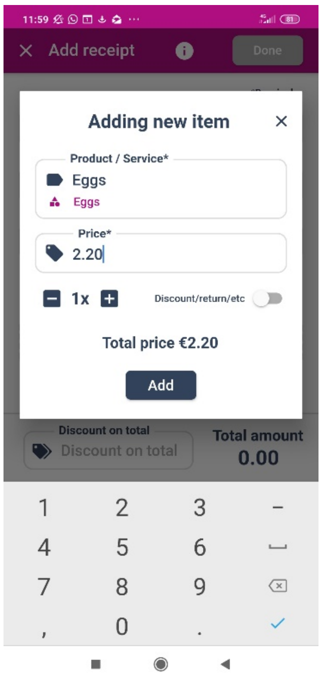


Mo	Tu	We	Th	Fr	Sa	Su
27	28	29	30	31	1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	1
2	3	4	5	6	7	8

Progress:
- 27 days completed
- 4 days missing
- -5 days remaining

Reward earned:
€27.00

Manual data entry:


Add receipt
Done

Adding new item

Product / Service*: Eggs
Eggs

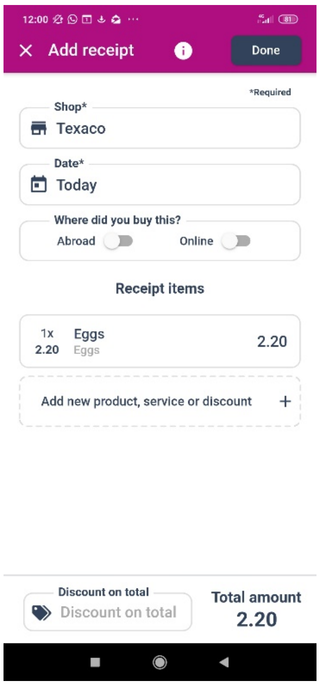
Price*: 2.20

1x
Discount/return/etc

Total price €2.20

Add

Discount on total: Discount on total
Total amount 0.00


Add receipt
Done

*Required

Shop*: Texaco

Date*: Today

Where did you buy this?
Abroad
Online

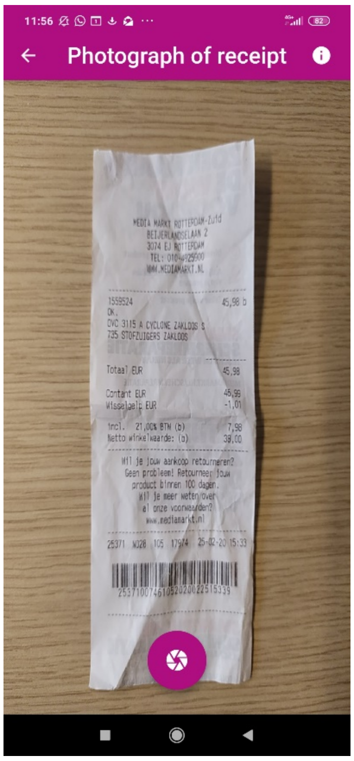
Receipt items

1x 2.20	Eggs Eggs	2.20

Add new product, service or discount +

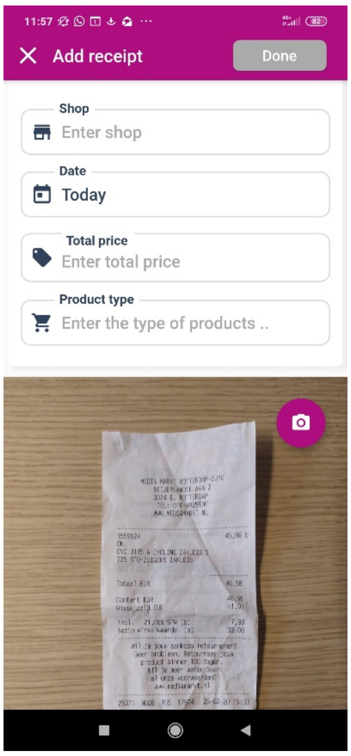
Discount on total: Discount on total
Total amount 2.20

Scan data entry:


11:56

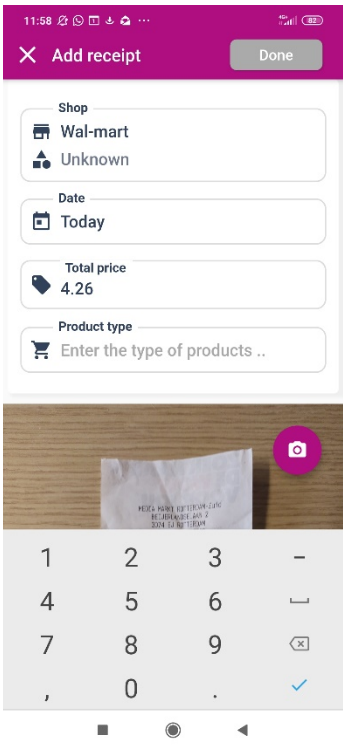
Photograph of receipt

```
MEDIA MARKT ROTTERDAM-Zuid
BEIJERLANDSELAAN 2
3074 EJ ROTTERDAM
TEL: 0104925900
WWW.MEDIAMARKT.NL
------------------------------------
1559524                      45,98 b
OK.
OVC 3115 A CYCLONE ZAKLOOS S
735 STOFZUIGERS ZAKLOOS
------------------------------------
Totaal EUR                     45,98

Contant EUR                    46,99
Wisselgeld EUR                 -1,01

incl.   21,00% BTW (b)          7,98
Netto winkelwaarde: (b)        38,00
------------------------------------
  Wil je jouw aankoop retourneren?
   Geen probleem! Retourneer jouw
     product binnen 100 dagen.
      Wil je meer weten over
      al onze voorwaarden?
        www.mediamarkt.nl
------------------------------------
25371  W028  105  17974  25-02-20 15:33

 25371007461052020022515339
```


11:57

Add receipt — Done

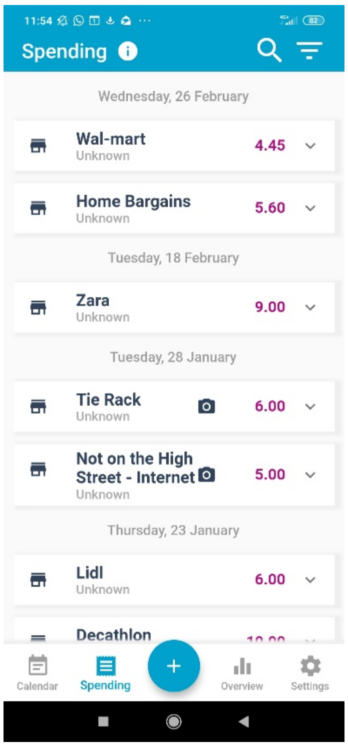
Shop	Enter shop
Date	Today
Total price	Enter total price
Product type	Enter the type of products ..


11:58

Add receipt — Done

Shop	Wal-mart Unknown
Date	Today
Total price	4.26
Product type	Enter the type of products ..

Expenditure overview and statistics overview:


11:54

Spending

Wednesday, 26 February

Wal-mart Unknown	4.45
Home Bargains Unknown	5.60

Tuesday, 18 February

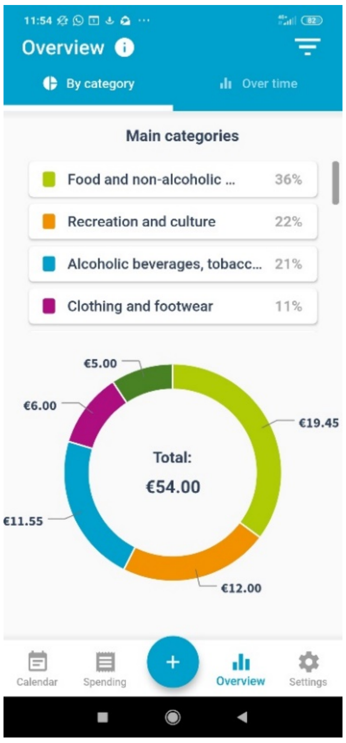
Zara Unknown	9.00

Tuesday, 28 January

Tie Rack Unknown	6.00
Not on the High Street - Internet Unknown	5.00

Thursday, 23 January

Lidl Unknown	6.00
Decathlon	10.00

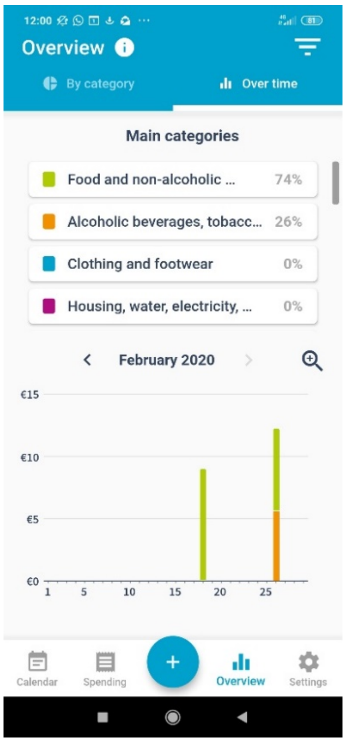

11:54

Overview

By category | Over time

Main categories

Food and non-alcoholic ...	36%
Recreation and culture	22%
Alcoholic beverages, tobacc...	21%
Clothing and footwear	11%


12:00

Overview

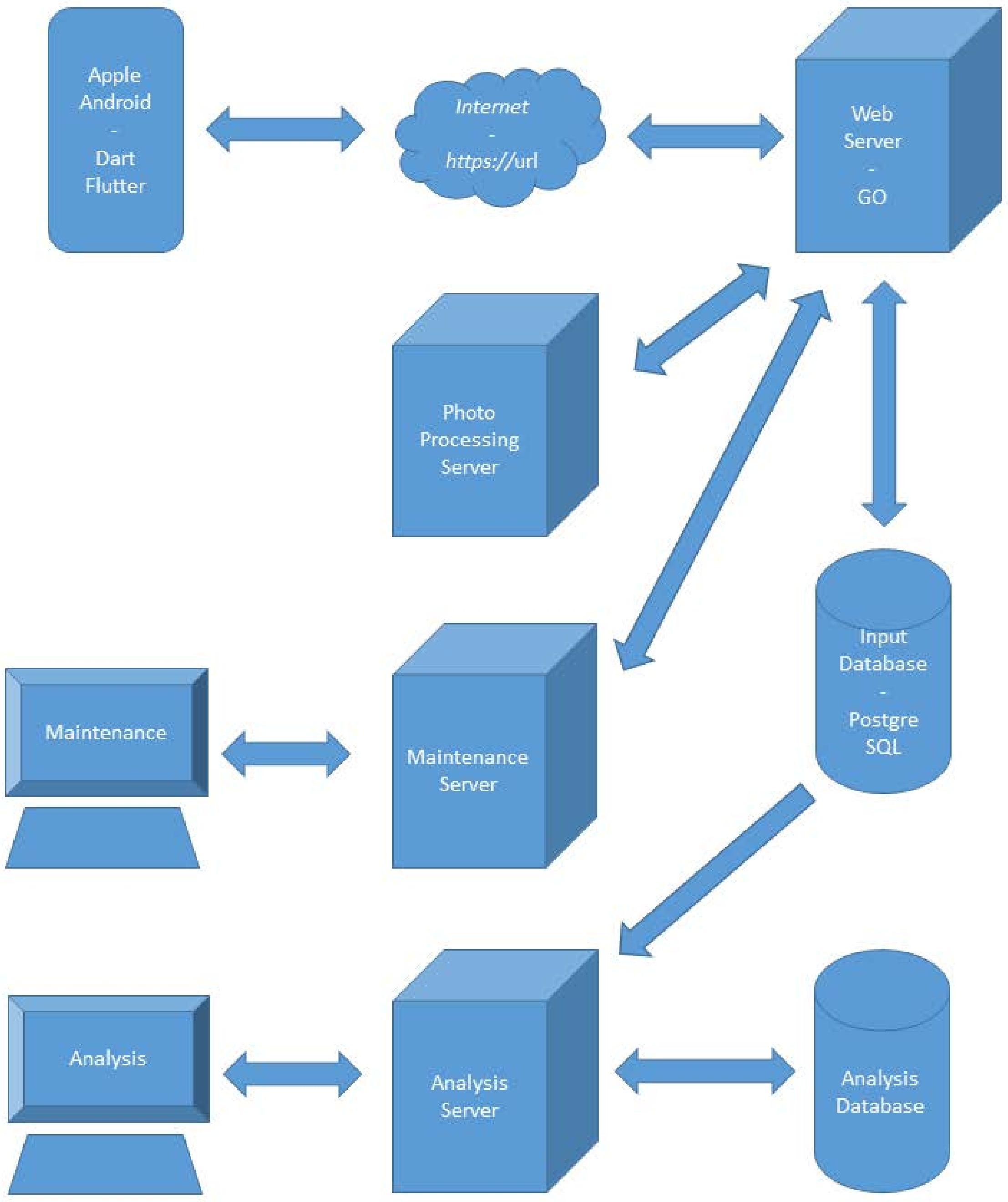
By category | Over time

Main categories

Food and non-alcoholic ...	74%
Alcoholic beverages, tobacc...	26%
Clothing and footwear	0%
Housing, water, electricity, ...	0%

ANNEX C – DETAILED DECSRIPTION OF BACKEND INPUT DATABASE

Household Budget Survey - Mobile App - System overview



System parts (implemented):

- Users enter/manage data on Apple or Android smartphones with HBS app. HBS app is developed in Dart/Flutter.
- Communication over internet via secure HTTPS URL.
- The Web Server monitors HTTPS URL and synchronizes data on smartphones with data in input database. Users can manage data on multiple smartphones. Data is timestamped on smartphone, data with latest timestamp is valid. The web server is developed in GO language.
- The Input Database stores and synchronizes data from smartphones. The database is also used for management of usernames and passwords (hashed). The input database is developed in PostgreSQL.
- The Photo Processing Server analyses photos of receipts. Optical character recognition is used to detect descriptions and prices of products. Descriptions in turn are linked to the Coicop classification. The resulting list of products is returned to the Web Server and Input Database and is synchronized with the phones of the user. At this moment the Photo Processing Server initiates the processing of receipts: it asks the Web Server for new photos to process. In the future the processing of receipts should be initiated by the Web Server: the Web Sever asks the Photo Processing Server to process a new photo.
- The Maintenance and monitoring of user response, management of passwords is accessible through a web browser.
- The Maintenance Server is a web server developed in Angular. It provides an user interface for the management of groups of users (passwords) and the monitoring of the usage of the HBS app by the users.

System parts vision (not yet implemented):

- Analysis of data by statisticians.
- Analysis Server for analysis of data in analysis database, and data import from input database to analysis database.
- Analysis Database for final data storage and data analysis.

Functionality

The main functionality of the HBS backend is:

- User authentication: check whether the user password matches the (hashed) password in the database
- Store receipt data: the data gathered on the smartphone is also stored in the database.
- Synchronize receipt data: keep the data synchronized between smartphones if the user uses multiple smartphones.
- Classification of receipt photos: find list of products on receipt photo (descriptions, prices and Coicop categories).
- Managing HBS app users: create groups of users and passwords.
- Monitoring: respondent activity and app usage is monitored and may be input to follow-up actions

The HBS-App maintains the receipt data locally on a phone. When the HBS-App is started, or when a receipt is entered or changed the app tries to connect to the GO server to synchronize the locally stored data with the data in the database. The HBS-App does not need a continuous connection with the GO server, when there is a connection all data will be synchronized.

Status

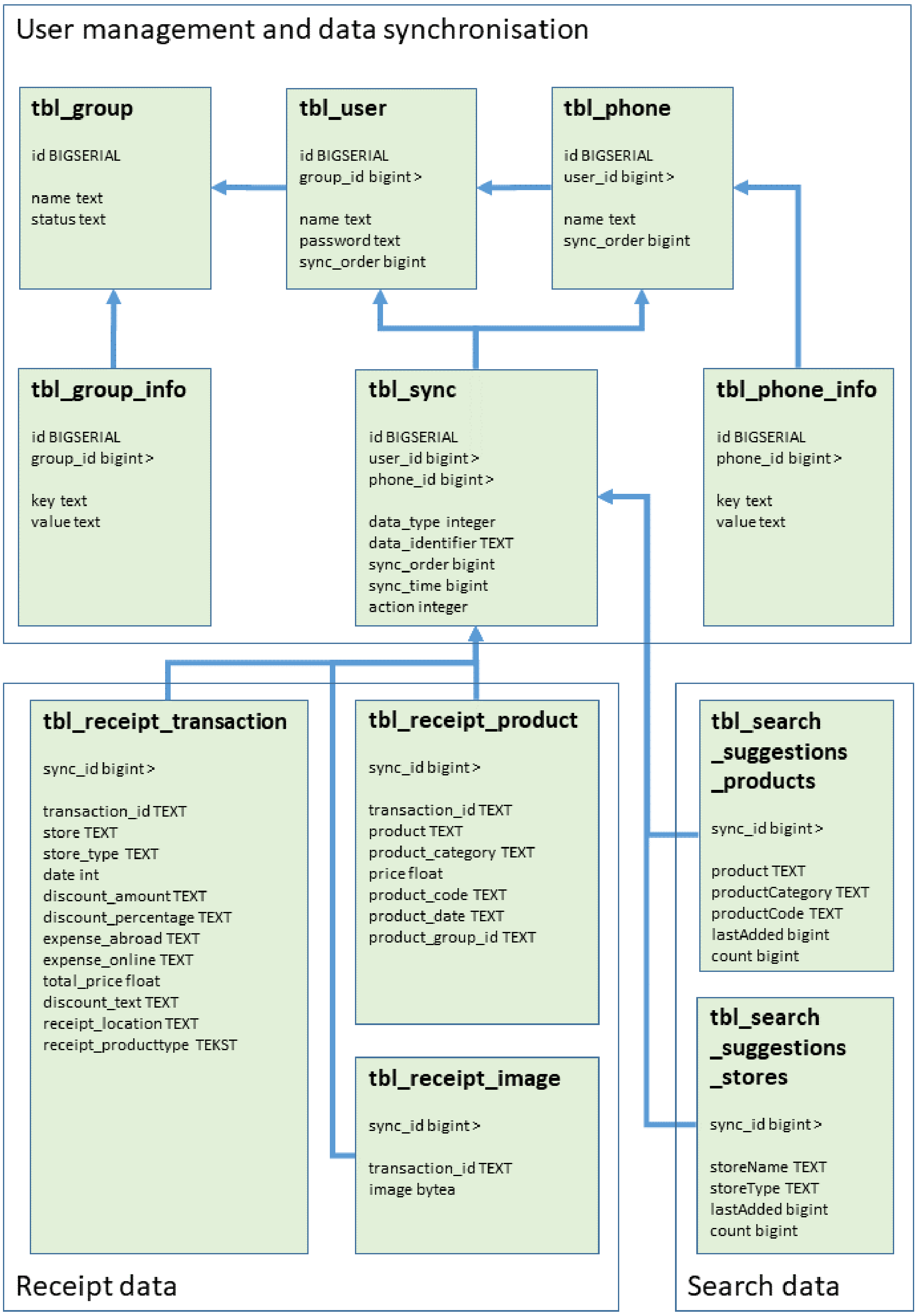
The HBS backend is operational but is still being developed further. Several countries tested a part of the system on the CBS infrastructure:

- HBS app
- HTTPS URL
- Web Server
- Input Database

At this moment we are preparing a large stress test for this part of the system. A small scale experiment showed that it can handle 100 receipt photos (250 KB per photo) in 15 seconds. This is without the Photo Processing functionality.
On a small scale we are also testing:

- Photo Processing Server (+/- 10 seconds processing time per photo)
- Maintenance Web Server (basic functionality)

Household Budget Survey - Data Model



PostgreSQL Input Database

The Input Database is implemented in PostgreSQL. It contains 11 tables (see data model above):

- tbl_group: survey group
- tbl_group_info: properties of survey group (controls functionality of HBS app)

- tbl_user: users in survey groups (hashed passwords)
- tbl_phone: phones of users (a user can have multiple phones)
- tbl_phone_info: properties of phones (information gathered on phones)
- tbl_sync: data synchronization mechanism
- tbl_receipt_transaction: receipt transaction (synchronized data)
- tbl_receipts_product: receipt products (synchronized data)
- tbl_receipt_image: receipt photo (synchronized data)
- tbl_search_suggestions_products: searched products (synchronized data)
- tbl_search_suggestions_stores: searched stores (synchronized data)

There is no further functionality in the database. The contents of these tables is managed by the Web Server.

The required size of the database depends on the number of users, the length of the survey period and the number of receipts a user enters per day. Per receipt the required database size is: +/- 5 MB, this is roughly the size of a photo + 1KB of textual data. Example: If 100 users take 2 photos per day for a period of one month, the required database size is: 100 * 2 * 5 MB * 30 = 30.000 MB = 30 GB. After a receipt photo is processed it could be removed from the database thus freeing up a lot of space (not implemented yet).

GO-Language Web Server
The Web Server is implemented in GO-Language. The file structure of the code is:

backend

- database

init.go	initialize database connection
tbl_<…>.go	insert, delete, update, select per table
data_<…>.go	data synchronization mechanism
daschboard_<…>.go	monitor user activity
create_tables.sql	sql script for database creation

- global

constants.go	global constants
functions.go	global functions
dashboard.go	types for dashboard
data_<…>.go	types for data synchronization
type_<…>.go	types other

- restapi

test.go	test if web server is up and running
group.go	manage groups
users.go	manage users
phone.go	register phones of users
process.go	photo processing
general.go	general methods
data_<…>.go	data synchronization
dashboard_<…>.go	monitor user activity
manual.txt	API manual

backend.go	main entry point https url

The API of the Web Server can be used to manage groups and users, to monitor user activities, to synchronize data over multiple phones and to access receipt images for photo processing.

The Web Server is configured via environment settings:

PORT	http port
DB_HOST	database host
DB_NAME	database name
DB_PORT	database port
DB_USER	database user
DB_PASSWORD	database password

Angular Maintenance Server
The Maintenance Server is implemented in Angular. It provides a user friendly interface to the API of the Web Server. At the moment we are testing a first basic version of this Maintenance Server, it can:

- manage groups and users
- generate a set of usernames and passwords for a specific survey group
- monitor survey progress (number of received receipts per user)
- show phone types used by users
- show receipt photos of individual user

The Maintenance Server is still under development. We also want to use it to monitor the usage of the HBS app (dwell times on pages, how often does a button get clicked), but for this purpose the HBS app has to be adapted (not implemented yet) to gather the required information.

Future steps
The not yet implemented part of the system overview is:

- Data analysis by substantive researchers and methodology
- Analysis server for analysis of data in analysis database, and data import from input database in analysis database.
- Analysis database for final data storage and data analysis.

While this functionality is not available the input database can be managed/queried directly with the pgAdmin_4 tool for PostgreSQL databases.

The HBS backend as it is, is equipped to authenticate users, store receipt data in an input database, manage groups of users, monitor survey progress and process receipt images. All parts of the HBS backend system will require further development/maintenance. Most interesting will be the Photo Processing Service. We are just starting tests to find out how accurate it can classify a photographed receipt (when the photographer is not holding a thumb over the receipt while taking a photo).

ANNEX D: GUIDE TO HBS WEB APPLICATION FOR MONITORING AND CASE MANAGEMENT

See attachment