

Vincent

Wine-focused questionnaire web application

Jan Polák – 2019-2020

Abstract

Vincent is a web application developed as a part of the Capstone project course at the Free University of Bozen-Bolzano for OENOLAB at unibz. It allows to create, manage, and fill questionnaires, with a special focus on questionnaires about wine quality, as perceived by trained panelists. The main reason for its development was the intent to discontinue the existing use of paper questionnaires, which are tedious to work with, and to transition to a paper-less electronic system. The application was developed over the course of one and a half months and is now ready for deployment.

Acknowledgements

My thanks goes to Simone Poggesi, whose input shaped the application, and also to the Prof. Dr. Emanuele Boselli who provided the initial design description. I'd also like to thank Marco Montali for consulting support.

Design requirements

As the initial project pitch solidified, the requirements of the project became well defined. The main core of the application is questionnaire filling, which is supported by a wide range of auxiliary systems, such as questionnaire definition templates and the unique account system. Since the whole system is not that large, no formal design document was necessary. However, many implicit design requirements were apparent and confirmed through customer consultation.

This section first describes the general functionality for perspective and overview, and then uses the PACT analysis methodology to introduce the general scope and context of this application's requirements.

General functionality

This is a short overview of the most significant features and functionality of Vincent, as described in this and other chapters of the report. Italics signify a common term of the application's vocabulary.

The application allows users to register. All regular users have to fill out a demographic questionnaire upon registration, for statistic and safety purposes. Those users with staff privileges can define questionnaire archetypes through a *template* system. *Questionnaire template* can be instantiated in a form of a *questionnaire*. Other users can be invited to the *questionnaire*. *Guest* users, without an account, can also be invited. Zero or more *wines* can be assigned to a *questionnaire*. A questionnaire with no *wines* behaves as if it had one anonymous *wine*. Each user will fill the *questionnaire* exactly once for each *wine*, in an order which is randomized per user. Each *wine* has a pseudorandom, three digit *code* attached to it, which is the only objectively identifying sign of the *wine* to the users which are filling out the questionnaire. Only the staff members know which *wines* correspond to which *codes*, to prevent bias.

When a *questionnaire* is ready (all wines have been added and participants invited), it can be *opened*. *Opened questionnaire* can be filled by the invited participants. Staff may now only invite more participants, modifications to the set of tested *wines* are no longer allowed. At arbitrary time, presumably at the end of the testing session, staff members can *close* the *questionnaire*. At this point, no further questionnaire filling is possible and no more participants can be invited. The last two significant actions are the result downloading, in the CSV format, and a *questionnaire* deletion.

Minor functionality of the application includes: a possibility to view all users in the system for the staff members, a password change mechanism and a password reset mechanism.

People

The application has two general types of users. Regular users, which can only fill questionnaires, and the staff, which also have access to the definition of the questionnaires and other management tasks. Regular users are further divided into two subgroups: the panelists and the guests.

A panelist is a regular participant of the wine testing process and has a special training, pertaining to wine quality assessment. For analytic, safety, and utility purposes, each panelist has to register and fill a form with various information. For example statistic features, such as gender, if they smoke, or from which area they originate, utility information, such as phone number, and safety features - intolerances to food or food additives. Each panelist also has assigned a numeric code, which is used when processing the questionnaire results. Each panelist has at most one account in the system.

A guest can be, for the most part, anybody. These users do not have a user account, but access the application through a unique URL link. (For technical reasons, the guest do have an account, so I will refer to their identity as a “guest account”, but the functionality of a guest account is very limited.) Guest accounts are created by inviting guests to a questionnaire and the accounts can only fill questionnaires. They do not strictly represent any one human being, as the account may never be used by anyone. If one person attends two different questionnaires as a guest, there is no requirement to use the same guest account.

Neither the panelists, nor the guests, can be assumed to have any technical skills beyond the ability to use a web browser and interact with simple webpages, so the application should be simple and should not present any novel user interface design decisions. Special care should be paid to the system’s accessibility, in case of movement or visual impairment of users. In the case of this implementation, the web technology should already be somewhat more accessible than paper, due to the existence of screen readers, page zoom and other assistive technologies. The implementation was created with these technologies in mind, though no specialized testing or adjustments were made due to time constraints and low overall priority of this feature. Special care was taken to ensure that the website’s elements will work correctly with keyboard controls.

The staff also have two different subgroups - the regular staff and the more privileged administrators. Their rights and capabilities within the system are mostly equal, with the exception that only administrators have full access to view the info on all registered users. This is done for privacy reasons, as the amount of people that may help with a questionnaire organization is potentially large, while the sensitive information about regular users should be kept as private as possible.

It can be assumed that the staff has at least some familiarity with the system and some technical capability. It can also be assumed that any would-be staff members can be trained

to be able to use the system effectively. That said, the system should still strive to minimize these costs, although the requirement is not as strict as with regular users.

Activities

All primary tasks related to this application are focused on questionnaire preparation, their filling, or result gathering for analysis. Secondary tasks are mostly uninteresting supporting tasks for the primary tasks, which exist solely to fulfill other design requirements, such as overall app security.

The application will be used only during and around the time of a questionnaire experiment. Most operations are not time constrained, with the exception of a special time-limited question type, which is described in later sections. Even though many people can use the application at the same time, no activities are done cooperatively by multiple participants.

Most management actions are deliberate, with a small risk of serious error, as most things can be manually reverted, with the exception of deletion, which is not revertible and may lead to a loss of data. This is partially mitigated by a confirmation dialog on all dangerous destructive actions and by marking these actions with danger-signalling visual design.

Questionnaire filling actions are mostly revertible, with a significant but important caveat. Questionnaires are divided into sections and it has been a strong requirement, that once a participant completes a section, it is no longer possible to go back to previous sections.

Currently all activities are about interactions with standard user interface elements or text.

Context

The interaction with the application, especially from the point of regular users, will physically take place in mostly controlled laboratory environment. This can be taken as given, because it is a requirement of the actual experiments themselves, to prevent any biases.

The interactions will not happen without context, but the users will often switch between their main focus - a glass of wine - and the application itself. In this way, the application is not a primary focus of panelists, but merely a tool in the background. Expected emotional states are both balanced and focused, though not necessarily on the application.

Technology

Even though initial design requirements mentioned a strong requirement to support mobile devices, the application does not manifest itself as a mobile app. This requirement is more easily met through a web application, which was designed with mobile devices in mind. The web ecosystem has an advantage over standard mobile applications that it is ubiquitous, very well suited to the client-server centralized architecture of this application

and that there is no need to re-implement same functionality for multiple platforms. It also has a wide support for accessibility technologies. The major disadvantage is that while it is supported everywhere, the support is not always full and various aspects may differ between browsers, especially visually. For this application I have targeted reasonably modern browsers. Anything younger than 5 years should have no problems and anything older should degrade gracefully, though this was not fully tested, because the scope is wide and the time was limited.

Implementation

The whole web application is implemented in Kotlin¹, which is a somewhat new (first stable release 2016) but popular² statically typed language running on the JVM and created by JetBrains (s.r.o.). It uses a few important software libraries, in decreasing order of size and importance:

- Undertow, a web server sponsored by JBoss
- H2 Database, an embedded relational SQL database, initially created by Thomas Mueller
- Exposed, a framework for safer and easier SQL database access, by JetBrains
- Kotlinx-html, HTML builder library for Kotlin, by JetBrains
- High Performance Primitive Collections by Carrot Search
- ICU4J, a localization toolkit by IBM
- Scrypt implementation by Will Glozer
- Guava, Java utilities by Google
- SLF4J, Logging interface
- tproll, Lightweight SLF4J implementation by me

The whole application is built by the Wemi build system, also of my creation.

The name of the application is Vincent, after the Saint Vincent of Saragossa³, martyr and the patron saint of winemakers, who lived in the 3rd century. However, for branding purposes, the webpage shows as SENSY.

The application was implemented in about 140 hours of programming, not including research and consultation (measured by a IDE plugin). It consists of 6500 lines of Kotlin code (not counting blank lines and comments). The total amount of lines of code over all technologies is just under 10 000.

¹ <https://kotlinlang.org/>

² <https://insights.stackoverflow.com/survey/2019#most-loved-dreaded-and-wanted>

³ https://en.wikipedia.org/wiki/Vincent_of_Saragossa

Architecture

The web application uses the traditional model of web requests and responses, not the “modern” JavaScript-driven AJAX dynamic model. This model was chosen for simplicity and for author’s mild dislike of JavaScript. In retrospective, this was a good decision, because the development of this kind of traditional web applications is indeed fairly simple. The dynamic AJAX model also has some benefits, but they are not significant for small and light websites. Typical page of Vincent has around 45kb without caching, and under 4kb with caching. For comparison, Google homepage has around 60kb with caching. (Transfer sizes, not decompressed sizes, which are larger.)

Internal architecture of the application is heavily dictated by the API design of the Undertow web server. Each HTTP request goes through a pipeline of handlers, most of which are standard handlers provided by Undertow for tasks such as input parameter parsing, HTTP reverse-proxy header handling, and so on (fig. 1).

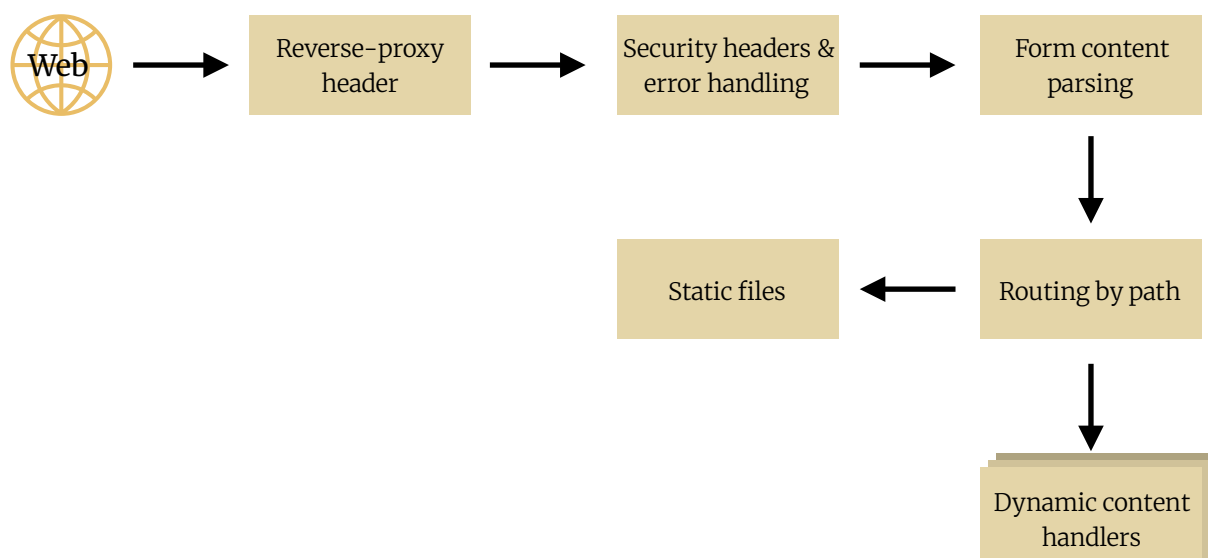


Fig. 1: Handler pipeline diagram

The pipeline handlers are asynchronous, which means that any operations that would otherwise block the thread doesn’t. This is a feature of Undertow and its default handlers, so there wasn’t a special effort to make it so. However, I have taken care to honor this behavior.

Routing handler implements a standard routing system. It picks the next handler using the path of the request, HTTP method, and other parameters. I am using the standard Undertow routing handler, with sub-handler registration that automatically takes care of some important global features. These features are: authentication checking (if required by the handler), idempotency checking (rejection of repeated requests, for example due to page refresh), and action predicate checking. Action predicate is a system for disambiguating request by their “action” form field. For example, both login and register form is handled through a POST request to the root path, but each has a different value in

the “action” field. The GET handler can also check if the account holder needs to complete the demography questionnaire and redirect the user there.

Rest of the internal architecture follows a simple functional (where possible) and procedural decomposition. Class methods are used very sparingly, though some parts of the code leverage Kotlin’s extension functions, which look similar to class methods. In general, the code benefits from not following standard OOP dogmas and instead focusing on simple functional decomposition.

Database design

The data is stored in a traditional SQL database over 8 tables (fig 2).

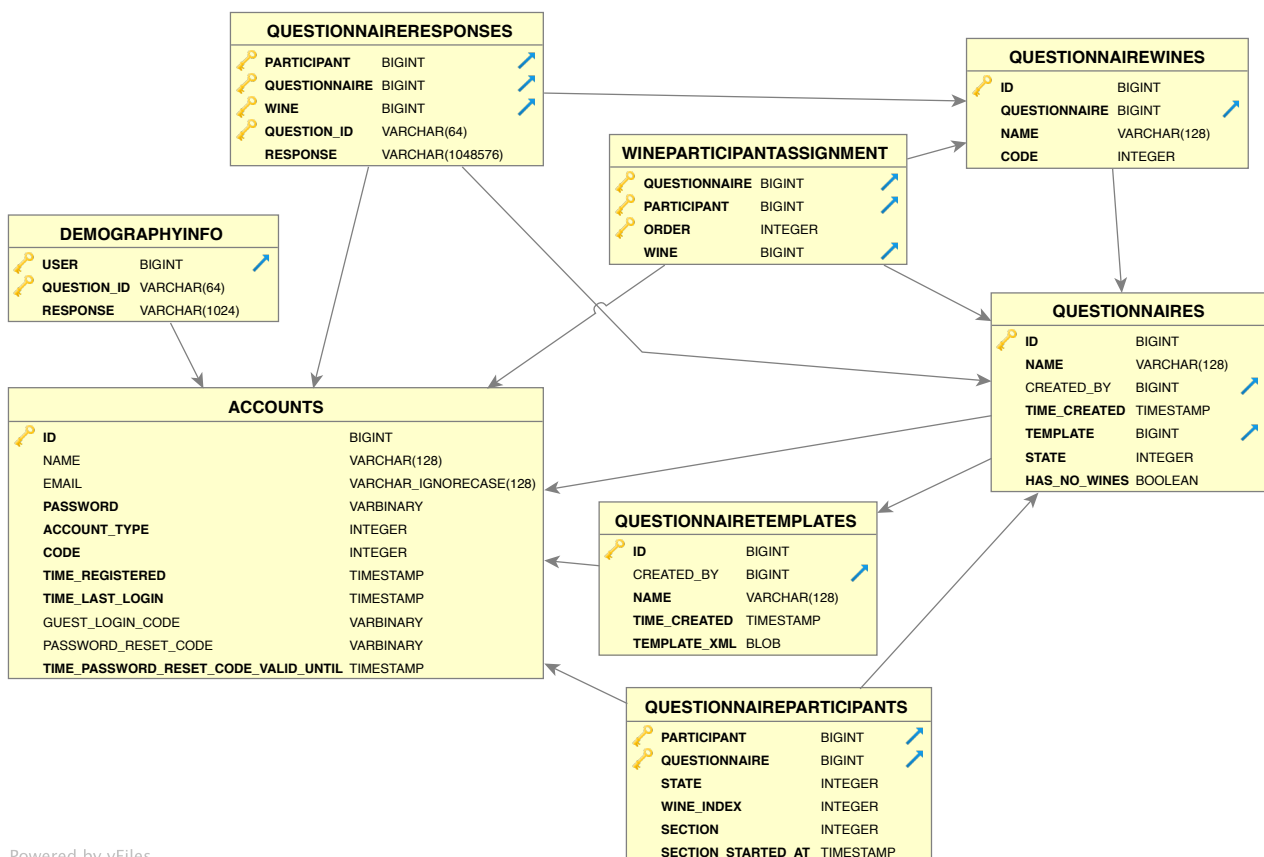


Fig. 2: Entity Relationship Diagram for Vincent’s database (generated by [DbVisualizer](#))

The database schema design is a compromise between ideal normalization and good usability. In particular, the only major problem is the duplication of information about questionnaire participant assignments over three tables: QuestionnaireResponses, WineParticipantAssignment and QuestionnaireParticipants. This duplication allows for a much simpler wine order and assignment logic. Not pictured are various indexes on columns which were added to ensure that all common queries will have an index to use. That said, no explicit performance testing was done due to low overall expected database load.

Security aspects

The application follows all major security guidelines related to these application types. The user passwords are stored under a secure password hash function Scrypt, with a unique salt per account. All sensitive public-facing database IDs are randomized or at least obfuscated. The application uses secure cookies, which are not accessible through insecure channels nor JavaScript. This requires deployment behind a reverse-proxy, which supplies TLS for the underlying HTTP protocol. An unsafe mode which does not require this is provided for testing and development, but should never be used in production. The application uses standard techniques to prevent CSRF⁴ and clickjacking⁵ attacks.

Questionnaire templates

For regular invited users, each testing session consists of filling one set of questions per each tested wine. This repetition is called (for the purposes of this application) a stage or a run. The set of questions itself is then defined by a questionnaire template, through a machine readable XML document with a well defined structure. The structure itself is described in detail by the application's internal documentation, easily accessible through the user interface, therefore this report will describe only the general essence of the system's possibilities.

All user facing texts of the questionnaire are fully localizable, if needed. This makes the markup somewhat verbose. To mitigate this for the common case of single-language questionnaire, the schema supports several syntactical shortcuts. Many questionnaire terms are also highly specific and must appear in the same language for all users, to prevent bias, while also showing the localized version for accessibility. This is also solved in the template system, by a marker which forces the text to appear in all languages.

Most user facing texts also support arbitrary embedded HTML. This provides formatting support and great extensibility, but presents a potential cross-site scripting attack surface. Questionnaire templates can only be created by staff members, which are hopefully trustworthy, so this risk seems acceptable.

Sections

The whole questionnaire run is divided into sections. Each section may have a title and zero or more questions or info blocks. These are presented on a single scrollable screen. Sections must be answered in order and it is not possible to go back to already submitted questions.

Some sections may not hold any questions, but may be there for other purposes. For example, a questionnaire will often present a welcome section, with a greeting and instructions for the testing session, similarly a goodbye section, and a spacer section,

⁴ https://en.wikipedia.org/wiki/Cross-site_request_forgery

⁵ <https://en.wikipedia.org/wiki/Clickjacking>

which should appear between any two wine runs with the purpose of reminding the participants to cleanse their palate and force a waiting period. For this purpose, a section may be configured to appear only on certain runs and may also force a minimum completion time.

Info blocks

These are, together with questions, the elementary building blocks of a section. They may contain a title and any arbitrary text. They are typically used to contain instruction text.

Questions

The most important part of a questionnaire. Like info blocks, they may contain title and instruction text, but also specify the type of the question, whether the question is mandatory, and the base name, under which the responses to this question will appear in the exported set of results. There are currently four types of questions that may be used. While many more question types are possible, these are the ones that are used in wine questionnaires.

Question type – One of

This question type allows the user to select exactly one option of the set of options. The available options may be arranged into categories (each with a separate title). Option may, upon picking, allow the user to specify further information, in the form of free text. For example, a question may have two options: “Without faults” and “With faults”. Picking “With faults” will prompt the user to specify, which faults the wine has.

Question type – Scale

While this question type is functionally a subset of the “One of” question type, as it allows to pick a numeric value from a whole-number scale with configurable range, it is an easy way to create a commonly used scale questions. It presents a simple distinct visual style, with configurable min and max labels. The numeric labels over each question are fixed.

Question type – Free text

As the name suggests, this question type allows the participant to answer using a natural language. The visuals can be customized to hint at the desired response type and scope – from a plain sentence, to paragraph and a single number.

Question type – Time progression

Some wine evaluation techniques call for a measurement of a variable, for example taste, over time. This question type provides this capability by presenting one of the first two question types repeatedly, with a customizable amount of time in between and the amount of repeats. It is unusual to require quick actions from users during a questionnaire

answering, so the implementation of this type has a strong focus on user feedback, to minimize any errors, while being forgiving to participants that can't keep up with the pace.

User interface & functionality

This chapter goes over all screens of Vincent and highlights unique challenges of each, while giving a sense of how they work. The presented version is without any additional branding, as it was originally designed.

A common font is used throughout the application - Merriweather, served from Google font service. This report uses the same font. Colors on all pages are chosen in such a way, to facilitate easy reading experience. The contrast ratios are kept at WCAG AAA levels.

Login & Registration

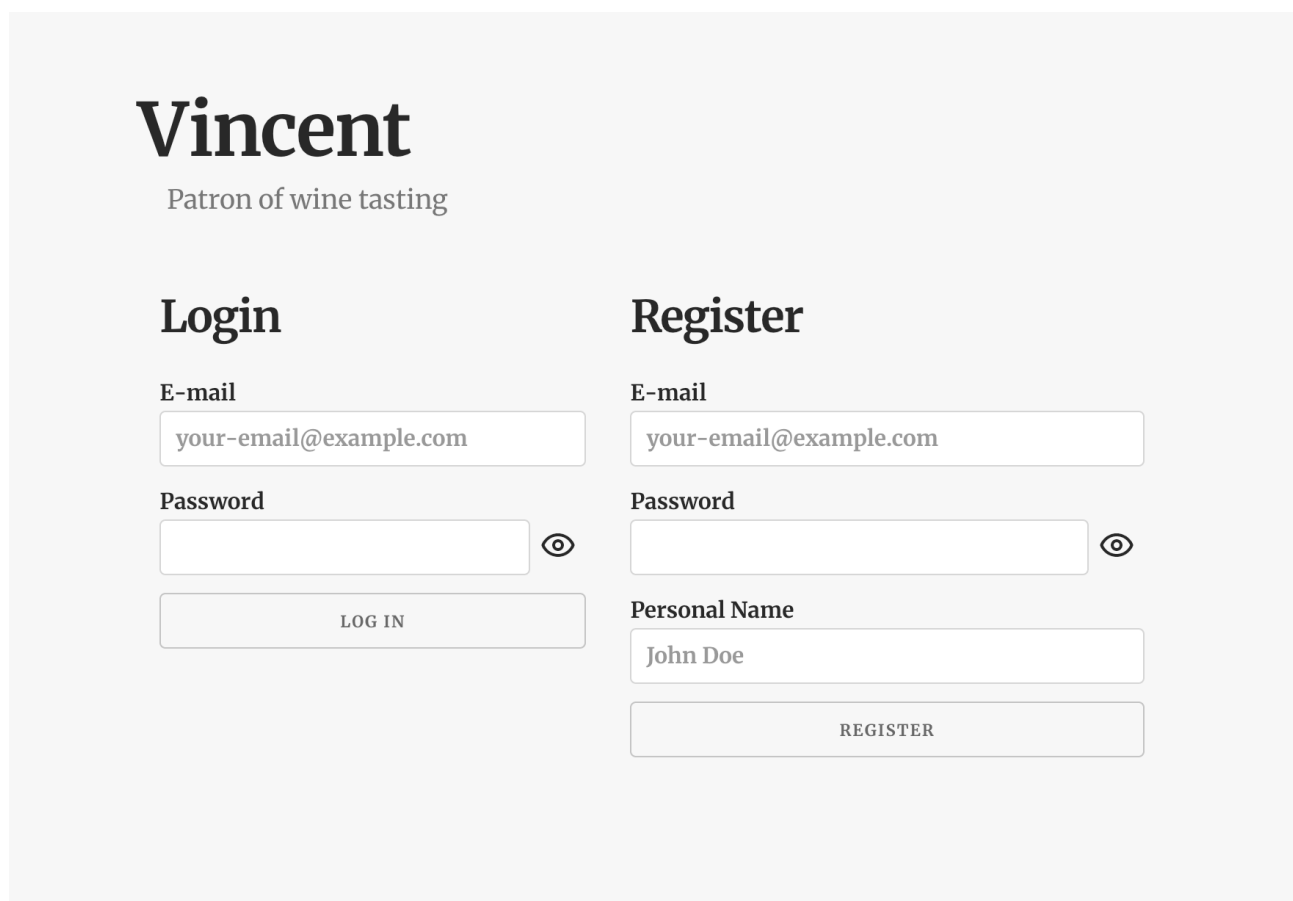
The image shows a web interface for 'Vincent', described as 'Patron of wine tasting'. It features two main sections: 'Login' and 'Register'. The 'Login' section has fields for 'E-mail' (with the placeholder 'your-email@example.com') and 'Password', followed by a 'LOG IN' button. The 'Register' section has fields for 'E-mail' (with the placeholder 'your-email@example.com'), 'Password', and 'Personal Name' (with the placeholder 'John Doe'), followed by a 'REGISTER' button. Both password fields include an eye icon for toggling visibility. The interface is clean and minimalist, using a light gray background and white input fields.

Fig. 3: Initial screen - Login and registration

This is the first screen that a user will see after visiting the webpage. I have decided to use an e-mail as the first login token, because most users don't have trouble remembering their e-mail addresses, as opposed to a generic username. The whole authentication process has been implemented in such a way, that is compatible with password managers.

The eye icon next to the password fields allows to temporarily display the password in plaintext, to check for errors. This technique is suggested by the section 5.1.1.2 of the NIST

Special Publication 800–63B about password handling. Most of the recommendations were followed, unless it was too technically demanding or time consuming.

Like all screens intended for (also for) regular users, this one is responsive. The two columns will stack upon each other, when needed.

There is a common system across the whole application, which is first visible on this screen. When a form which has associated buttons is fully filled, the button(s) will turn green through a CSS animation, inviting the user to click it.

Demographic questionnaire

VINCENT John Doe

Hello!
Before we start, please fill out the following questions.

All information collected will be kept as sensitive and protected data

Phone number OPTIONAL
Use only for sensory analysis session

Gender
Female Male Other

Year of birth
1973

Home country
Italy

Home country region OPTIONAL
If commonly used.
Alto Adige - Südtirol

Highest completed education
See [Wikipedia](#) for more information about ISCED 2011 education levels.

None Primary (ISCED 1, Scuola Elementare) Secondary (ISCED 2 or 3, Scuola Media/Superiore)
Post-secondary (ISCED 4 or 5) Bachelor or equivalent (ISCED 6, Laurea)
Master or equivalent (ISCED 7, Laurea magistrale) Doctoral or equivalent (ISCED 8) Higher

Do you smoke?
No Yes

Do you have any [food intolerance](#) or [food allergies](#)?
No Yes

Do you have sulfite intolerance?
Sulfites have a number of technological functions, including antioxidant, bleaching agent, flour treatment agent and preservative, and are used in a wide variety of applications in the food/wine industry. The Health World Organization committee established for sulfite of 0-0.7 mg/kg bw per the limit. The level added in the wine it's below this limit. For more information [click here](#) or see [the Wikipedia article](#).

No Yes

FINISH

Fig. 4: Demographic questionnaire (split screenshot)

Upon first login (typically right after registration), the user is presented with a demographic questionnaire. The underlying visual style is the same as the style of wine questionnaires. The initial “Hello” text is present only when opening the questionnaire for the first time - when subsequently opened for response revision from the profile screen, the text is just a plain “Personal info” header.

The questions in the questionnaire follow the questions used in the paper version of the same document, albeit slightly modified. Modifications were done to make the questions less assuming, more inclusive and more precise. Additional information was added to the last two health related questions, to make sure that everybody fully understands the meaning of the question, as any errors here could have serious consequences.

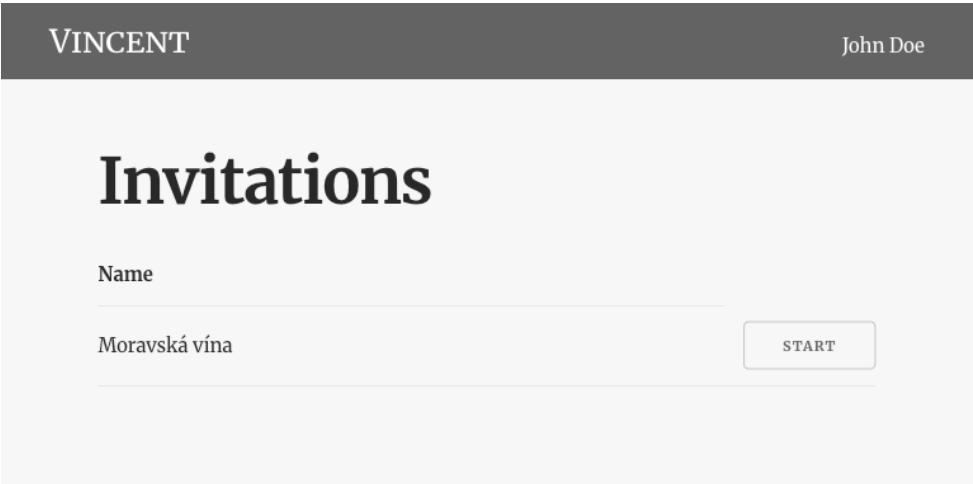


Fig. 5: Home screen for regular users

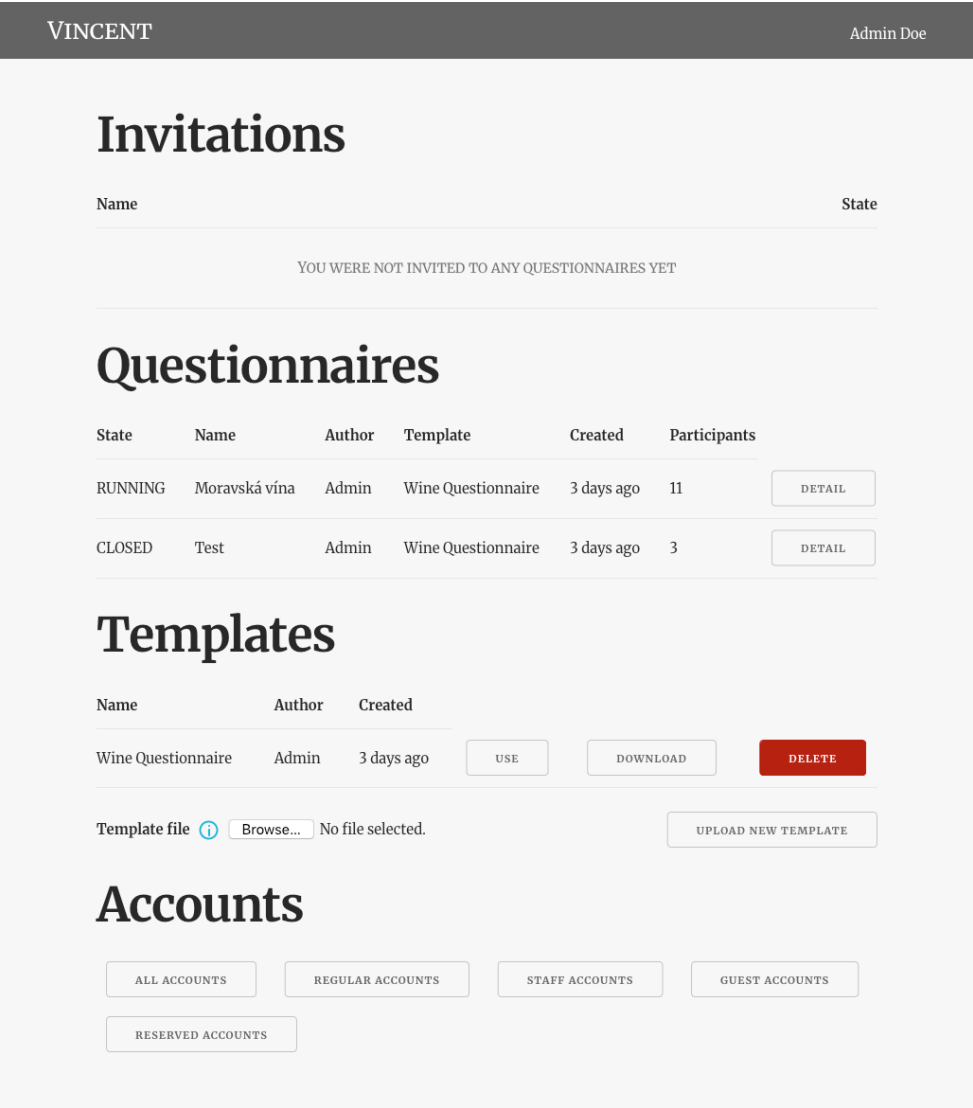


Fig. 6: Home screen for staff users

The home screen is the main entry-point to most of the functionality and serves as a central menu and informational dashboard. There are four main sections.

Invitations

This is the only section that is also accessible to regular users. It lists all questionnaires, which are currently open and which they can participate in. Only the questionnaire name and a button to start (or continue) answering is shown.

Questionnaires

In this section, the administrators can view all currently existing questionnaires in the system and go to their detail screens. The questionnaire creation itself is done from the next section, which would suggest that this section should be after it, not before. But since I expect this section to be more actively used the other one, this order makes more sense.

Templates

The questionnaire templates live here, along with options for their creation, deletion and downloading. It is also possible to create a new questionnaire using the given template. The blue information button in the new template upload menu leads to a comprehensive documentation of the template XML schema, the basics of XML itself and a small section about HTML inline formatting, to the extent that will be useful in preparation of questionnaires.

Currently in development is the option to replace the XML of a given template with a different one. This is an inherently dangerous operation when the template is being used, unless the question schema itself is unchanged, because it could lead to an inconsistent response collection. For this reason, this operation is only available to administrators. It can be very useful for last minute changes in questionnaires, for example to fix typos, formatting problems, change wording, etc. The need for this functionality was identified during the alpha testing session.

Accounts

This section consists of a collection of buttons, which all lead to the account list page, with different filtering settings. This page is further described in its own section.

Future expansion

During the preparation for the alpha testing session, it became apparent that a new functionality, which will likely warrant its own section, will be needed. It is the need to be able to upload and host arbitrary files (mostly images) to be used in the questionnaires.

Questionnaire detail

VINCENTAdmin Doe

Moravian Wines

Wine Questionnaire

Guest invited

Participants

#	Name	E-mail	Code	Sulfite Intolerance	Food Intolerance	State
1	Jane Doe	jane@jane.com	102	Yes	No	Invited
2	John Doe	john@john.com	101	No	No	Invited

Invitees

ex@mp.le, 123, ...

INVITE

Guest Participants

#	Guest Code	Login URL	State
1	G#746D0C8D26782E95	http://localhost:7000/guest?g=G%23746D0C8D26782E95&auth=TJjFlveA09ryv8unYq00ZQ==	Invited
2	G#1EE056DFACDD6415	http://localhost:7000/guest?g=G%231EE056DFACDD6415&auth=1EpN_dEBZvdYPKJpb_gifg==	Invited

Amount

1

INVITE GUESTS

DOWNLOAD LINKS

Wines

#	Name	Code
1	Merlot	957
2	Pálava	198
3	Tramín	892
4	Tramín (repeat)	238

Wine name

Chardonnay

Code

Random

ADD WINE

Participant Wine Assignment

#	G#746D0C8D26782E95	G#1EE056DFACDD6415	102 Jane Doe	101 John Doe
1	892 Tramín	238 Tramín (repeat)	892 Tramín	892 Tramín
2	198 Pálava	892 Tramín	238 Tramín (repeat)	957 Merlot
3	957 Merlot	957 Merlot	957 Merlot	238 Tramín (repeat)
4	238 Tramín (repeat)	198 Pálava	198 Pálava	198 Pálava

OPEN THE QUESTIONNAIRE

DELETE QUESTIONNAIRE

Fig. 7: Questionnaire detail page – created state

All aspects of a questionnaire can be changed on this page. Some aspects can be changed only when the questionnaire is in a particular state - for example it is not possible to add more wines after the questionnaire has been opened.

The page opens with the questionnaire's name, which is modifiable. Then it lists regular participants, including their intolerances (if logged as administrator). This section contains also a form for invitation of registered and guest participants. Participants can be specified by their panelist code, email, full name or guest code (guest participants only). Multiple participants can be specified at the same time by separating their identifier with a comma.

Next section contains the overview of invited guest participants, with the ability to invite more in bulk and the ability to download the list of all login URLs of all guest participants.

Next two sections focus on the wines. First is the list of all wines added to the questionnaire, with the ability to add more. Wine codes can be pre-selected or generated randomly. They can also be changed manually later. What follows is an informative section which contain the information about the order in which the participants will test given wines. The cells of the table have their background colored according to the wine code, for easier orientation. The colors were chosen with a high emphasis on differentiability, even for those with color blindness or other vision impairment.

At the bottom of the page, there are buttons to open/close the questionnaire, to delete it, and to download the results, when they are ready.

Questionnaire answering

VINCENT

Jane Doe

Palate

WINE: 892

Body

Low 1 2 3 4 5 High

☐ ☐ ☒ ☐ ☐

Alcohol

Low 1 2 3 4 5 High

☐ ☐ ☐ ☒ ☐

Sweetness

Not sweet Sweet Very sweet

☐ ☒ ☐

Acidity

Low 1 2 3 4 5 High

☐ ☐ ☒ ☐ ☐

Astringency

Low 1 2 3 4 5 High

☐ ☒ ☐ ☐ ☐

Balance

Unbalanced Well-Balanced

☐ ☒

Retronasal

Fruity rust with nuts

Length/Finish

Short Medium Long

☐ ☐ ☒

Condition

No fault With fault

Faults/Taints Rusty

NEXT

VINCENT

Jane Doe

Appearance

WINE: 892

Clarity

Presence of sediments Absence of sediments

☐ ☒

Color tonality

White
Lemon Yellow Gold Amber
Rosè
Pink Salmon Copper
Red
Purple Ruby Garnet Tawny

☐ ☒ ☐ ☐ ☐

Color intensity

Pale 1 2 3 4 5 Deep

☐ ☐ ☐ ☒ ☐

Fluidity

Low 1 2 3 4 5 High

☐ ☐ ☒ ☐ ☐

NEXT

VINCENT

Jane Doe

Nose

WINE: 892

Condition

No fault With fault

Faults/Taints Oxidized

Intensity

Low 1 2 3 4 5 High

☐ ☐ ☒ ☐ ☐

Aromas

Fruity rust

Complexity

Low 1 2 3 4 5 High

☐ ☐ ☐ ☒ ☐

NEXT

Fig. 8: Questionnaire answering – example questionnaire

Content of this page depends on the questionnaire template. One full example can be seen in the fig. 8. The question types and other features are described by the earlier section on questionnaire templates.

Account list

VINCENT														Admin Doe
All accounts														
Name	E-mail	Account type	Code	Registered at	Last login at	Food intolerant	Sulfite intolerant	Phone number	Gender	Year of birth	Home country/region	Education	Smoking	
?	?	Guest	G#746D0C8D26782E95	2020-02-16	?	?	?	?	?	?	?	?	?	DELETE ACCOUNT
?	?	Guest	G#1EE056DFACDD6415	2020-02-16	?	?	?	?	?	?	?	?	?	DELETE ACCOUNT
Admin Doe	admin@admin.com	Admin	100	2020-02-16	2020-02-16	?	?	?	?	?	?	?	?	
Jane Doe	jane@jane.com	Normal	102	2020-02-16	2020-02-16	No	Yes	?	Female	1973	Brittania - Island of Man	Master or equivalent (ISCED 7, Laurea magistrale)	No	RESET LOST PASSWORD DELETE ACCOUNT
John Doe	john@john.com	Normal	99	2020-02-16	2020-02-16	No	No	?	Male	1920	Czechia - Karlovy Vary	Bachelor or equivalent (ISCED 6, Laurea)	Yes - 1	RESET LOST PASSWORD DELETE ACCOUNT
Number 1	1@example.com	Normal	150	2020-02-16	2020-02-16	No	Yes	?	Female	1985	France - Paris	Doctoral or equivalent (ISCED 8)	No	RESET LOST PASSWORD DELETE ACCOUNT
Number 2	2@example.com	Normal	151	2020-02-16	2020-02-16	Yes - Gluten	No	?	Male	1990	Spain - Valencia	Master or equivalent (ISCED 7, Laurea magistrale)	No	RESET LOST PASSWORD DELETE ACCOUNT

DOWNLOAD AS CSV

DELETE UNUSED GUEST ACCOUNTS

E-mail

mail@example.com

Code

000

ASSIGN CODE

Fig. 9: Account list – All accounts

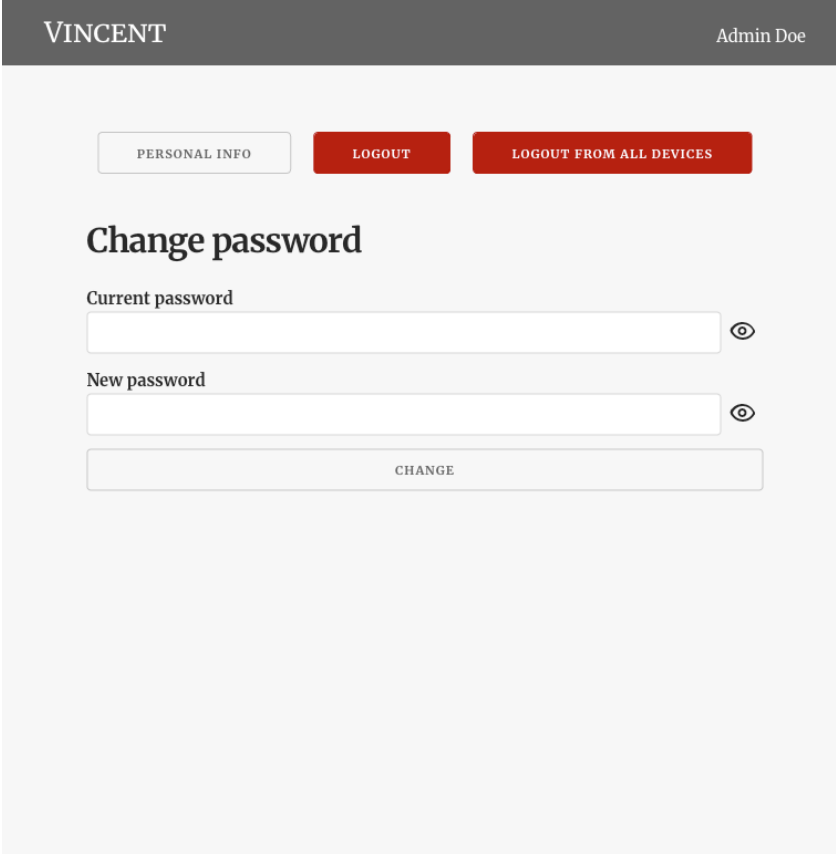
This page displays all accounts, that fit certain criteria, or, as is the case on fig. 9, all accounts. All information in the column “Registered at” and further columns to the right is only visible to administrators, not regular staff, as it is sensitive. Different filtering modes expose different column sets – for example when showing only guest accounts, the demographic data is not shown (as there is none for the guests). Staff members cannot see any information about administrators, for security reasons, only about other staff members.

Several actions are available (mostly to administrators). Per user (without special privileges), there is account deletion, and resetting of lost passwords. The password is reset to a random strong password, which is displayed to the resetter. This is not ideal from security standpoint, but given the trustworthiness of administrators and the low security importance of these accounts, it was deemed acceptable. The alternative – sending “forgot password” reset e-mails was also considered, but not implemented, because sending e-mails is not trivial from the development perspective and it would be hard to deploy correctly.

There is also the possibility to download the whole list as a CSV and to delete all unused guest accounts. This last action is necessary, because guest accounts are often used only for a single questionnaire, and often in large numbers. This would clutter the interface and database. Guest account is considered unused, when it is not invited to any questionnaire.

The last function on this page is the ability to reserve or re-assign a panelist code. This feature was specifically requested to ease the migration from the existing paper-based system. It makes most sense to use this function after a database reset. This feature can also be used to pre-create accounts for people that will attend a testing session and need to be invited, but haven't registered yet. The code may be changed freely, but no two accounts may share the same panelist code. Guest accounts cannot have a panelist code - the immutable guest code is used instead.

Profile



The screenshot displays the 'VINCENT' profile page for 'Admin Doe'. At the top, a dark header bar contains the name 'VINCENT' on the left and 'Admin Doe' on the right. Below the header, there are three buttons: 'PERSONAL INFO' (light gray), 'LOGOUT' (red), and 'LOGOUT FROM ALL DEVICES' (red). The main content area is titled 'Change password' in bold. It contains two password input fields: 'Current password' and 'New password', each with a toggle icon (an eye) to the right. Below these fields is a 'CHANGE' button.

Fig. 10: Profile page

The profile page's purpose is mostly personal account administration. Buttons allow to return to the personal info questionnaire and review responses and to logout. The "Logout from all devices" button is only available to administrators, to prevent confusion of regular users. The login sessions are implemented using session cookies, which are not long-lived, so it is not a very important feature anyway.

There is also the possibility to change the account's password.

This page is reachable by clicking on own name in the top bar. Clicking on the header logo ("Vincent" in this case) leads back to the home screen.

Command line interface

Command line interface is not a screen of the user interface per se, but it is an important part of the application, especially for the administrators. It supports a few simple commands.

The most commonly used one can change a privilege level of an existing account. This is the only way to create new staff member and administrator accounts. Another command can change a password of an arbitrary user, in case they forget (the password reset function available to administrators can be only performed for regular accounts, for security reasons).

Third command is a bit more interesting - it allows to reserve a panelist code for a user which registers with a given e-mail. This feature is analogous to what is available on the account list page to administrator and uses the same program code internally.

Lastly, there is a command to gracefully shut down the whole application server.

The server launcher itself can be configured using standard command line arguments. Both of these command and configuration systems have help information available, at the usual places.

Alpha testing session

An alpha testing session to evaluate the usability and stability of the whole system took place on 14. 2. 2020. The preparations for the session and the session itself highlighted a need for several features, which were either promptly implemented, or are still being worked on. These features are mentioned in their relevant sections throughout the report, but were usually small or small-medium sized. All missing features were fairly low-impact, typically just quality of life improvements. No critical functionality was absent.

The testing itself went smoothly, with only very minor problems, which were usually not caused by the software itself, such as small mistakes in used templates. I am pleased to say that professor Emanuele Boselli found the application “impressive”. Feedback collected from users was limited, but there were no complains and one user praised that the software was easy to use.

Conclusion

The wine evaluation questionnaire application Vincent, developed as a part of the Capstone project course was successfully created and serves its intended purpose. This was verified during a testing session with real panelists. The application is now ready to enter into production environments.

Several areas of further development were identified, for example a custom user interface for aroma selection, using the aroma wheel theory, or a special variable-over-time tracking systems using a joystick. These ideas still need more specification effort before they become implementable. Nevertheless, they fall out of scope of this project in the perspective of the Capstone project.

It was interesting and educational to work together with scientists, and it was satisfying to be able to help with solving their day-to-day problems, hopefully allowing them to focus their time on more important problems.

Table of contents

Abstract	1
Acknowledgements	1
Design requirements	2
General functionality	2
People	3
Activities	4
Context	4
Technology	4
Implementation	5
Architecture	6
Database design	7
Security aspects	8
Questionnaire templates	8
Sections	8
Info blocks	9
Questions	9
Question type - One of	9
Question type - Scale	9
Question type - Free text	9
Question type - Time progression	9
User interface & functionality	10
Login & Registration	10
Demographic questionnaire	11
Home	12
Invitations	13
Questionnaires	13
Templates	13
Accounts	13

Future expansion	13
Questionnaire detail	14
Questionnaire answering	16
Account list	17
Profile	18
Command line interface	19
Alpha testing session	19
Conclusion	20
Table of contents	21