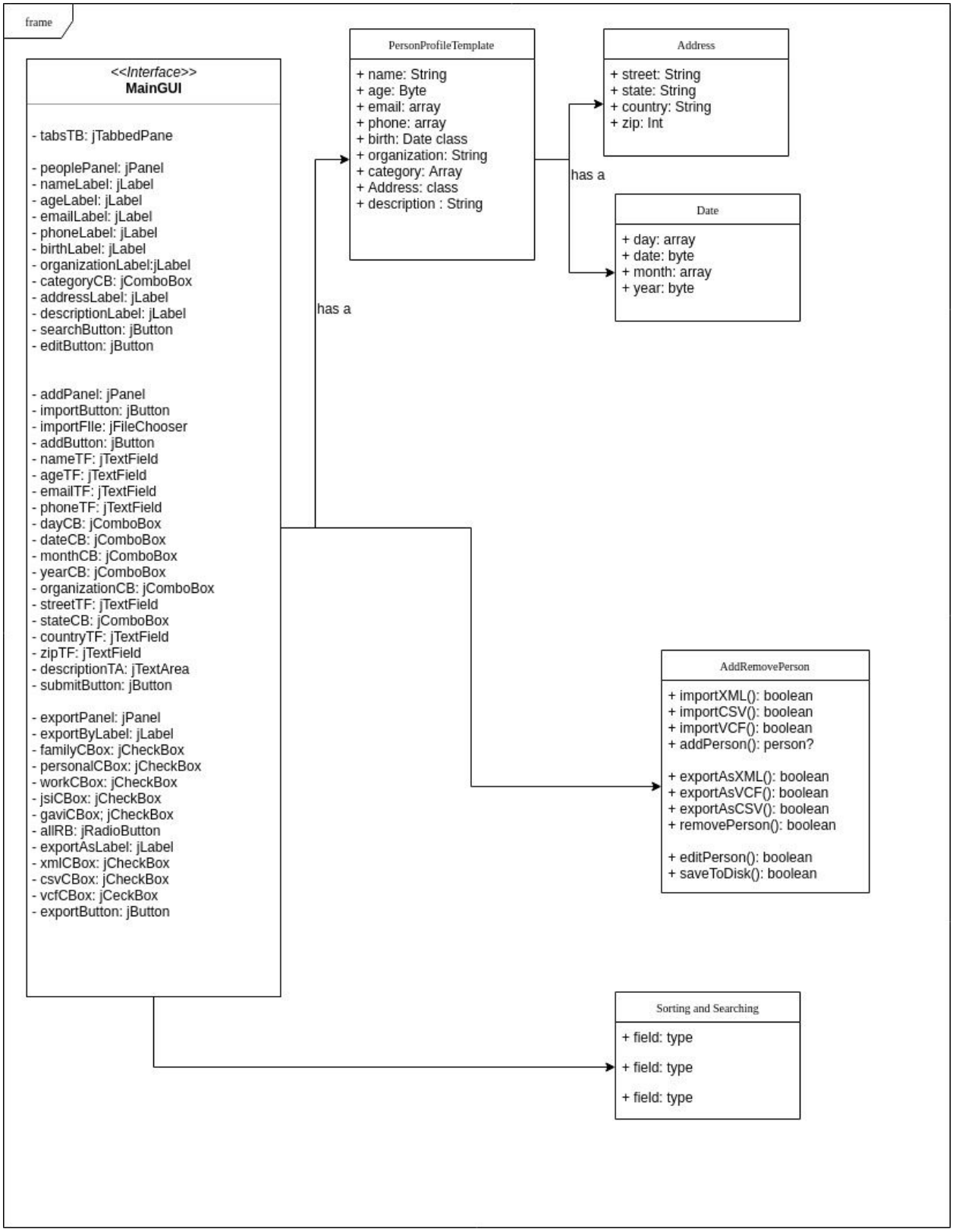
Input Table

Input	Type	Example	Border	Extreme
Name	String	Kaiser Wilhelm II	Non-ascii chars, very long inputs	No input
Age	int	51	Decimals, fractions, non-int chars	
Date of Birth	String	1 November 1914	Non-ascii chars, very long inputs	
Emails 1, 2, 3	String	bismarck@airmail.cc		
Phones 1, 2, 3	String	+66 123 456 7890		
Organization	String	Weyland-Yutani Corporation		
Address	String	34 crane street Arkham, Mas, US, 12345		
Categories	String	Family		
Notes	String	Social media profiles, etc		
Imported Contacts	.vcf	workcontacts.vcf	Very large files, files that contain duplicates, empty files, unreadable/corrupted files	
Imported Contacts	.csv	oldcontacts.csv		
Save File	file	personArrayList.txt		

Output Table

Output	Type	Example
Save File	file	personArrayList.xml
Exported Contacts	.csv file	familycontacts.csv
Exported Contacts	.vcf file	personalcontacts.vcf

UML Diagram


frame
<<Interface>>
MainGUI
- tabsTB: jTabbedPane
- peoplePanel: jPanel
- nameLabel: jLabel
- ageLabel: jLabel
- emailLabel: jLabel
- phoneLabel: jLabel
- birthLabel: jLabel
- organizationLabel:jLabel
- categoryCB: jComboBox
- addressLabel: jLabel
- descriptionLabel: jLabel
- searchButton: jButton
- editButton: jButton
- addPanel: jPanel
- importButton: jButton
- importFIle: jFileChooser
- addButton: jButton
- nameTF: jTextField
- ageTF: jTextField
- emailTF: jTextField
- phoneTF: jTextField
- dayCB: jComboBox
- dateCB: jComboBox
- monthCB: jComboBox
- yearCB: jComboBox
- organizationCB: jComboBox
- streetTF: jTextField
- stateCB: jComboBox
- countryTF: jTextField
- zipTF: jTextField
- descriptionTA: jTextArea
- submitButton: jButton
- exportPanel: jPanel
- exportByLabel: jLabel
- familyCBox: jCheckBox
- personalCBox: jCheckBox
- workCBox: jCheckBox
- jsiCBox: jCheckBox
- gaviCBox; jCheckBox
- allRB: jRadioButton
- exportAsLabel: jLabel
- xmlCBox: jCheckBox
- csvCBox: jCheckBox
- vcfCBox: jCeckBox
- exportButton: jButton
PersonProfileTemplate
+ name: String
+ age: Byte
+ email: array
+ phone: array
+ birth: Date class
+ organization: String
+ category: Array
+ Address: class
+ description : String
Address
+ street: String
+ state: String
+ country: String
+ zip: Int
has a
Date
+ day: array
+ date: byte
+ month: array
+ year: byte
has a
AddRemovePerson
+ importXML(): boolean
+ importCSV(): boolean
+ importVCF(): boolean
+ addPerson(): person?
+ exportAsXML(): boolean
+ exportAsVCF(): boolean
+ exportAsCSV(): boolean
+ removePerson(): boolean
+ editPerson(): boolean
+ saveToDisk(): boolean
Sorting and Searching
+ field: type
+ field: type
+ field: type

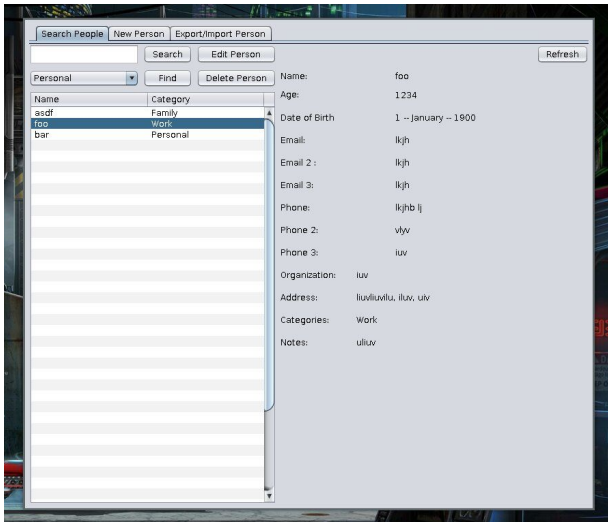
Prototyping Process

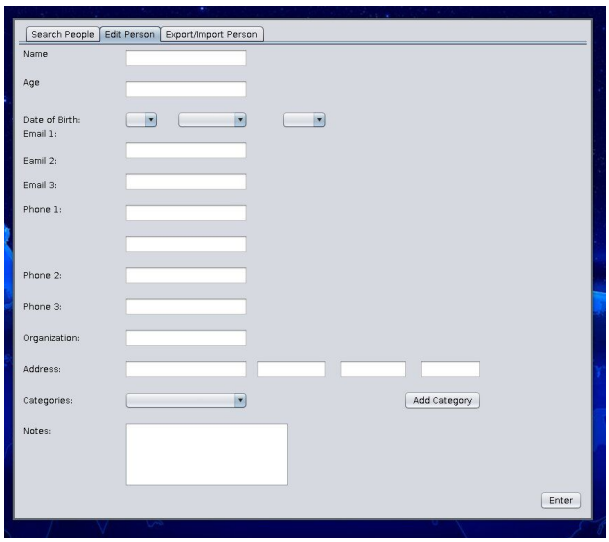
Unfortunately I lost the original prototype GUI, but these are the changes I remember making following the second interview and on into further development. In keeping with the feedback of my client I added sorting features and removed the search button in favour of a field, all of which was initially moved to another searching tab.

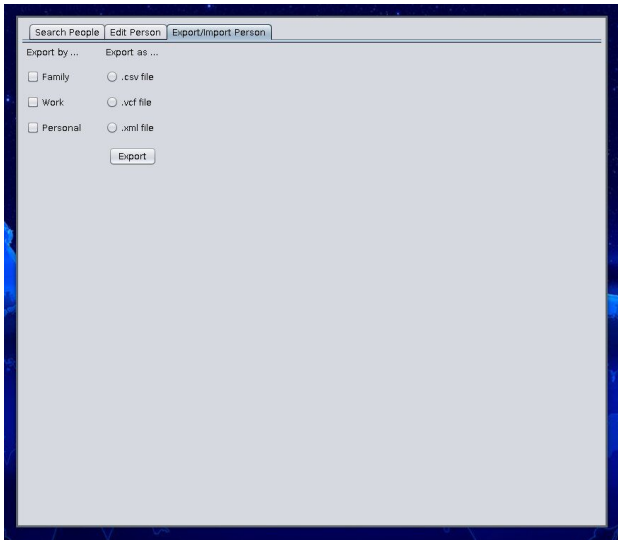
I later reduced the number of sorting features and restored the search button, believing that on-the-fly searching would be too hard to implement. I decided to go from an add button on the People tab to a separate tab for editing and adding people. In order to reduce the UI's complexity, I also brought the search and sort features back into the main tab, which I renamed Search People.

As my client suggested I created separate fields for different parts of the address, however they are not labeled.

The final reworked prototype:







Chronological Development Plan

1. Build final GUI in MainGUI
2. Write template classes
    a. Person
    b. Date
    c. Address
3. Create basic database
    a. Initial peopleArrayList
    b. MouseReleaseEvent to create new instance of Person
4. Complete interface between code and GUI
    a. Write contents of ArrayList to peopleTable
    b. Display attributes of selected Person on peopleArrayList
5. Write SearchSort class
6. Add database features
    a. MouseReleaseEvent to remove instance of Person
        i. Create confirmation dialogue box
    b. MouseReleaseEvent to edit instance of Person
    c. MouseReleaseEvent to search peopleArrayList
    d. MouseReleaseEvent to query peopleArrayList
    ~~e. MouseReleaseEvent to edit categories~~
        ~~i. Create dialogue box~~
7. ~~Import and export .csv and .vcf files~~
8. ~~Read and write database contents to and from a file~~

---> NOT PART OF THE WORD COUNT <---

Appendix -- Prototype Feedback in Second Interview

The GUI is clean and well-organized, with placement of the fields in logical order common to other contact applications. The simplicity is a change from commercial applications that put an emphasis on design, but this does not impact the ease of usability of the GUI. There are appropriate fields for essential contact information, I like the 'People', Import', and 'Export' tabs at the top and the three action buttons (Search, Add, Edit Profile) below. Everything is visible and self-explanatory.

The Search feature should be a field instead of a button, which sorts as the user types. Also, the contact fields should include a category, linked to a sort feature, with standard categories like family, friends, work, but enabling user-defined categories, too. The sort feature will then be used to sort by category or by other fields, such as email domain.

Also, please change the Address field to include standard address fields like street address, city, state/province, postal code,country, and include more than one phone field option (home,work, mobile).

Another field might include Skype, WhatsApp and other messaging options. And maybe social media like FB and LinkedIn?

Word Count: 244