# A Level Computer Science Project Writeup

# Analysis:

## Problem Identification

I am going to make a **Invoice software** app named **“reddere”** which is latin, so adds sophistication and is pronounceable by many languages as many languages are latin derived. This will be a software that you can enter products and/or services into and it will **calculate and display as a PDF** invoice, which can **later be emailed** to the client.

I will use computational methods such as, **decomposition** to break down the code, so it’s simpler and easier to understand the sections I need to create. I will do this by breaking down the frontend and backend and emailing system into different files. I will use kivy which helps to compartmentalize a modular design and make it easier to create and understand the layout, and the code for future use.

Another computational method i will be using is **Abstraction.** This method will be used to remove all extra information, for example background images, and it means I can focus on completing the app and then adding in the additional functions afterwards. Removing unnecessary information and functions.

I will also use methods like the **Trial and Error approach** to complete my project, as finding what works and what doesn’t will make my life easier moving forward.

I will use methods of **Client feedback** to ensure that the app I am creating is the app that is wanted by the **Stakeholders** as getting my target audience involved will help make the app better.

* The problem is the lack of Invoice Software that is available open source and is easy to install and run. I am building a python-based Invoice creation app that has PDF functionality.
* I will use decomposition to break down the app into sections, modularization to compartmentalize the project into different segments. I will use Trial and Error for creating the app components, allowing them to be tested in different scenarios.

## Stakeholders

This app is aimed at Sole Traders and small business owners who need a simple and easy to use UI to complete simple invoicing tasks. It can ofc be used by bigger businesses that do not require many functions, but require an app that works well for invoicing. However the aim will be 1->5 people companies that want simplicity. I will create tutorials to help those of all ages, from experienced company owners to young teenagers just starting out. The backgrounds and icons will be simple and openly sourced, making the UI functional and removing the need for me to create artwork and / or find colours that work (I am colourblind). This also allows it to be simple and not distracting.

* This is aimed at Sole Traders and small business owners, it will target UK based companies, due to tax calculations and locale limits. The requirements are that of a low footprint app that doesn’t take up lots of space on a device. It Primarily targets Windows Computers.

## Research

In my app, the user needs to enter products and services and use templates to format a document, I have included some similar apps and services that do this, and have given me inspiration for my solution.

Invoicehome is a mobile and browser-based invoicing software that is easy to use and has simple templates for you to fill in with forms. The downsides are that it limits the number of invoices you can create behind a paywall, which doesn’t work for bigger companies, or Sole Traders with many invoices needed to be created. It requires a login and is hard to customise the branding and colours of the templates. The upsides are that it is cloud-based and therefore can be accessed from anywhere, and that it can do quotes alongside invoices. The main upside is the use of templates so that all you need to do is fill out the forms with your data and it will populate the template. From this i would take the ability to create quotes, and the idea of templates is good, but I would want more customisable templates or custom templates all together. I definitely don’t want to include the paywalls but the ability to email from within the software may be nice to include.

InvoiceHome Demo.png

Canva is a browser based design software that uses drag and drop sections for design, I like this as I could use it for the template editing, an upside to this is that it is easy to customize the template, but a downside of Canva is the need for a browser and the internet being consistent to create something, so i would take from this that i need to create a system that does not constantly rely on an internet connection to create an invoice, only needing it to send it out if wanted. I would include the ability to add images and text boxes, and the usage of templates. I wouldn’t include the account, and limits on the creation of images and decals.

Canva Demo.png

Photopea is a free, web-based image editor designed to mimic the functionality of Adobe Photoshop, offering tools for tasks like photo editing, graphic design, and web graphics creation. The interface closely resembles Photoshop, making it familiar for experienced users. Accessible directly from the browser without installation, Photopea provides a powerful alternative for professionals and hobbyists, with both free and premium features.

Photopea Demo.jpg

Essential Features:

The software needs to have the ability to create PDF invoices, that can then be sent to clients. This means it must be formatted nicely. It must be easy to use and intuitive to use the app. - This means easily recognisable symbols and menu labels. It must have the ability to add custom branding, and/or customise the look of te invoice. This means custom Logo’s, custom colours and/or custom templates. The invoice must work for both products and services.

Limitations: The limitations of my app is that it may only be able to export the PDF invoice, as building sharing functionality within the app is not a priority. Mobile use won’t be as good as desktop usage, as the sizings of the GUI and invoice will be designed for desktop computers / laptops. The app won’t be able to take payments, but will have the ability to add bank details, as to be able to use a BACS transfer for payment.

Hardware & Software Requirements: As there is no movement or complicated computation functions, limited overhead is needed. These are the minimum recommended requirements: For the hardware requirements of my app, the user will need to have at least 2.5Ghz, this is because the average is 3Ghz of CPU speed on modern computers, so setting the minimum requirements to below the average allows older devices to run the app at optimal speeds. 2Gb of RAM is recommended as the app will serve the webpage from RAM, it allows plenty of room for the app to run, and all the additional content (extra branding files) to be shown. Python must be install on the system, all app functionality has been tested up to 3.12.7, however other versions may work, with the required packages installed (as outlined in the README of the project). These should be the listed versions to ensure compatibility with each other. The device must have an operating system installed so that the user can interact with python. The device must have an operating system to show the Invoicing software An internet connection is recommended to download the app, however it is not required for running of the application. The user may require external hardware, such as a keyboard to input the details, mouse as to interact with the setup GUI and screen to interact with the app.

# Design

## Decompose the problem

User Interface: - Input forms (customer info/branding, Product/Service details) - View Invoices - Generate Invoices - Edit/Update Invoice Details - Navigation Menu (Home, Settings, Invoice History)

Invoice Management: - Create Invoice - Generate unique invoice ID - Add Products / Services - Add Customer Details - Save Invoice - Update Invoice - Delete Invoice

Product / Service Management: - Add New - Update (Pricing, description) - View - Delete

Reporting: - Generate invoice reports (total sales, outstanding payments) - Filter reports by date, customer or product - export reports (PDF, CSV)

Settings: - Configure Tax Rates - Customise template - Currency settings

## Design of Structure

### Flowchart:

Design - Flowchart

### Create a decomposition diagram and explain and justify what you have done and why you have done it.

Design - Decomposition Diagram

### Create pseudocode of your program – you will probably do this. Explain what you have done and why you have done it. You will add pseudocode for your different versions for top marks.

[Pseudocode.md](./Pseudocode.md)

### Sketch / draw screens of your game – (not actual game/program) / program and label

Version 3 - Dev Sketch

## Usability Features

The menu will work as pages of the app, with the nav menu always being shown, no matter what page your on. This adds continuity for the user, and intuitive designs make apps easier to use. The display will be divided into sections, with each one having a title that explains it, borders may be used to make it obvious where each section starts / ends. The controls will be easy to use, as there will be a call to action for submitting the data, and filling in the information will be simple. Visual clarity, e.g. using greyed out zones for unnecessary data, and highlighted zones for important, required information. There is no need for additional styling, as the app will be functional and easy to use. The flexibilty of using custom Templates for invoices, gives the user more freedom to style as wanted. There will be lots of additional data validation to reduce any errors.

## Data Dictionary Table

| Variable name | Size / Value | Datatype | Description | Validation |
| --- | --- | --- | --- | --- |
| Invoice Number | 000001 | String | Invoice ID number | 6 Digits |
| Currency\_Code | GBP | String | International currency code | 3 Digit Currency code |
| Provider[Bank\_Account\_Number] | 12345678 | Integer | Provider’s Bank Account Number | Must be 8 chars |
| Customer Hanndler Email | sales@customer.com | string | Customer Sales Handlers email | Must be in the format x@y.z |
| Product Title | Product1 | string | Product Title | Must be plain-text string |

## Validation

Data will be validated as it is parsed, this restricts what data can be inputted where. This will be done through data types, e.g. making sure that only integers are entered as the invoice number, and that only A-Z, a-z, are entered as names. The menu will be greyed out on the current page, stopping any reloading of the current page.

## Iterative Test Table

| Test No. | What testing & why | How testing | Test data (What is entered) e.g. left key pressed | Expected result |
| --- | --- | --- | --- | --- |
| 1 | Does the program start with no errors | Run the main python file | N/A | A cli for FLASK should begin, and a localhost invoice site should be created at 172.0.0.1:8000 |
| 2 | Can you input data for use in the invoice | Either a GUI pops up or user input is prompted for data. | All Products, Services, Customer Attributes | Data inputted appears on the invoice, e.g. products & contact details |
| 3 | Data validation exists | Input wrong data into data input | “QWERTYT1” inputted as InvoiceNumber | Failed as data should’ve been integer type |

## Post Development Testing

~~Same table as above but this final testing including testing of each~~ **~~success criteria~~** ~~and each~~ **~~usability feature~~** ~~from design that you will test when the program is finished.~~ ~~Done after each version.~~

## App Version 1:

App Version 1.png Positives: - Invoice can be created and shown, using a web browser (so no external app must be used, e.g. PDF reader) - Calculates subtotal & Total with given factors - Shows Contact Information - Show Date & Calculates Due Date

Improvements: - Make Logo Image Show Up - Be Able To Differentiate Product & Service - Have a section for each? - Add QR Code as link to the page - Print To PDF - Email PDF - Make Due Date timings Adjustable (not always 14 days) | Test No. | What was tested | Did it succeed | why? | | — | — | — | — | | 1 | App starting | Yes | It opened as intended | | 2 | Data input | No | Although not able to input data by users, data inputted in program | | 3 | Data validation for InvoiceNumber | No | Able to be any datatype. |

## App Version 2:

App Version 2.png Positives: - Logo image now shows up - Products and Services are now defined and shown seperately - PDF can be downloaded

Improvements: - Enter data in each time (and keep some data persistant) - Logo File Upload - GUI Menus for entering data

| Test No. | What was tested | Did it succeed | why? |
| --- | --- | --- | --- |
| 1 | App starting | Yes | It opened as intended |
| 2 | Data input | No | Although not able to input data by users, data inputted in program |
| 3 | Data validation for InvoiceNumber | No | Able to be any datatype. |

## App Version 3:

App Version 3 - Dev Sketch.drawio.png App Version 3.png Positives: - Has a intuitive GUI to input data. Negatives: - Not all functions work - e.g. Templates page

| Test No. | What was tested | Did it succeed | why? |
| --- | --- | --- | --- |
| 1 | App starting | Yes | It opened as intended |
| 2 | Data input | No | Although not able to input data by users, data inputted in program |
| 3 | Data validation for InvoiceNumber | No | Able to be any datatype. |