

Submission 1 Team Project

Mockup

Below is the screen which displays the customer's expenses.

The mockup shows a web browser window titled "Expenses" with the URL "https://www.website.com/expenses". The page features a search bar labeled "Search Expenses..." and a date selector "dd / mm / yyyy" with a calendar icon. A central box displays "<< Website Name >>". On the right, it shows "Monthly Expenses: \$ __" and "Annual Expenses: \$ __". The main content area contains eight rounded rectangular buttons arranged in two rows. The first row includes "Fixed Expenses" (Regular recurring and does not change), "Variable Expenses" (Can change from month to month), "Necessary Expenses" (Essential to maintain basic standard of living), and "Discretionary Expenses" (Not essential and is a luxury spend). The second row includes "One Time Expenses" (Occur infrequently and only once), "Savings & Investment" (Saving money or investing in assets), "Debt Repayment" (Paying off debt such as loans or credit card payments), and "Other Expenses" (Expenses which are unique). At the bottom left is an "Export" button, and at the bottom right is a help icon (question mark).

Figure 1

Figure 1 portrays the expenses screen in the web application. This screen will allow the customer to view their previous expenses. They can either decide to view all their previous expenses or specific expenses under a certain category. For instance, if they specifically want to view their *Discretionary Expenses*, they can simply navigate to the screen displayed in Figure 1 from the home screen and then click on this button to be directed to a different screen interface. This will show them a visual representation of all their previous *Discretionary* expenses in the past 6 months. They can click on another to view their data from 6 more months back. Moreover, they can also search for a specific expense using the search bar located in the left top corner of the screen illustrated in Figure 1. The *Export* button, is also provided should the customer wish to export their data in a format that can be used for tax purposes or for import into other financial software. Every screen of the web based app also contains a help button located in the right bottom corner explaining to the user the purpose of the particular interface they are currently on.

Persona

Below is a user persona outlining 1 customer's profile.

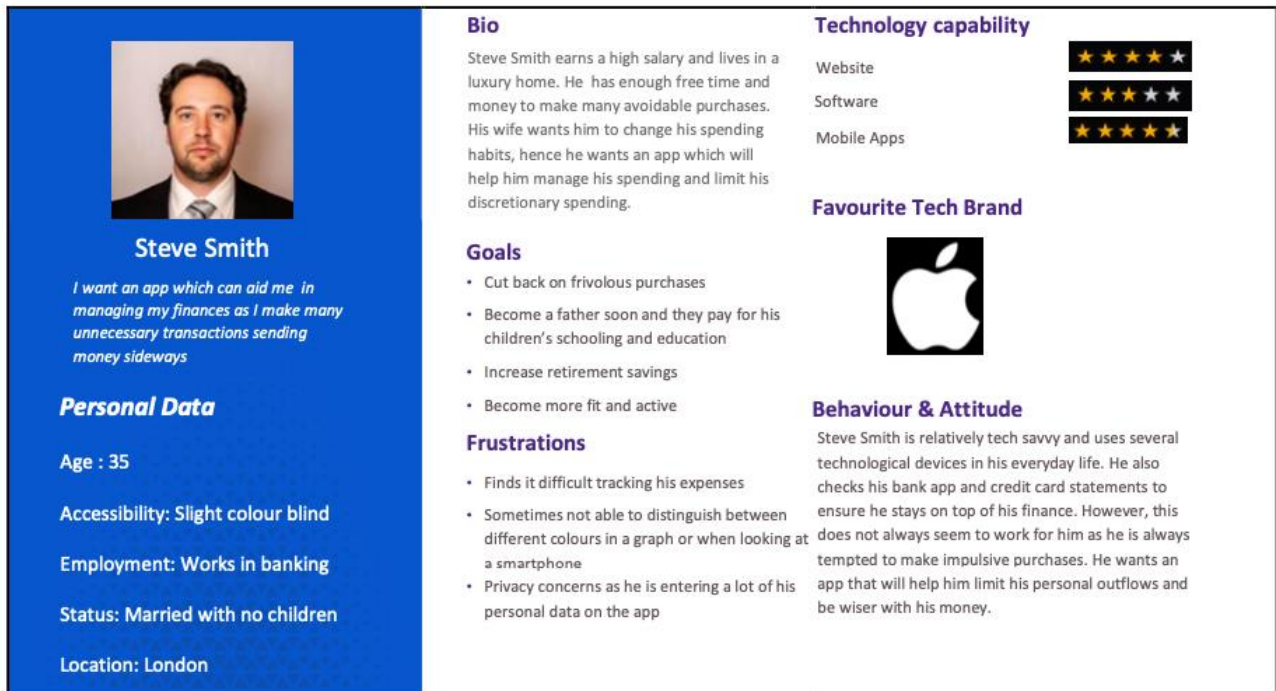


Figure 2

Outlined in *Figure 2* is 1 customer's data we have collected. Steve Smith is a potentially a customer who fits our target demographic. He is someone who requires help in managing his expenses as his cash outflow is higher than he would like. He wants an app which will prevent him from making impulsive rash decisions which he then regrets later. He is slightly colour blind so sometimes he is not able to distinguish between different colours, especially if they have a similar hue, hence he wants this app to accommodate his requirements.

Kanban Cards

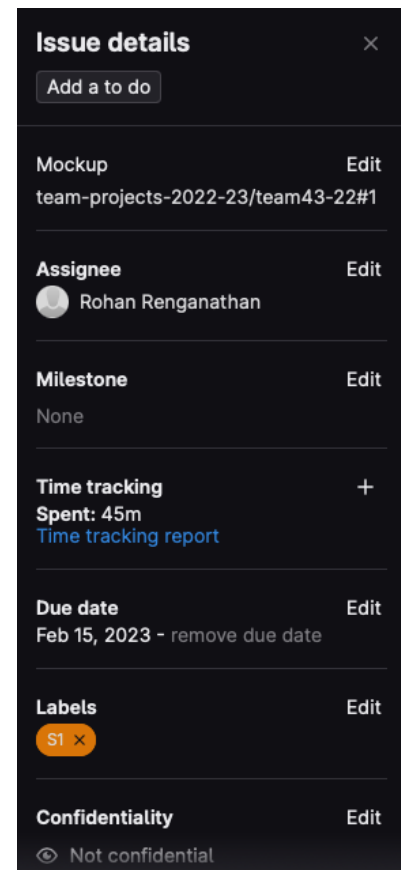


Figure 3

Figure 3 depicts the first Kanban card I created. This is a mockup for the *Expenses Screen* of the web based finance tracker app. This is labelled under S1 as this comes under Submission 1. It is also not confidential to allow other members in my group the chance to look at it and provide any suggestions or improvements that could be made.

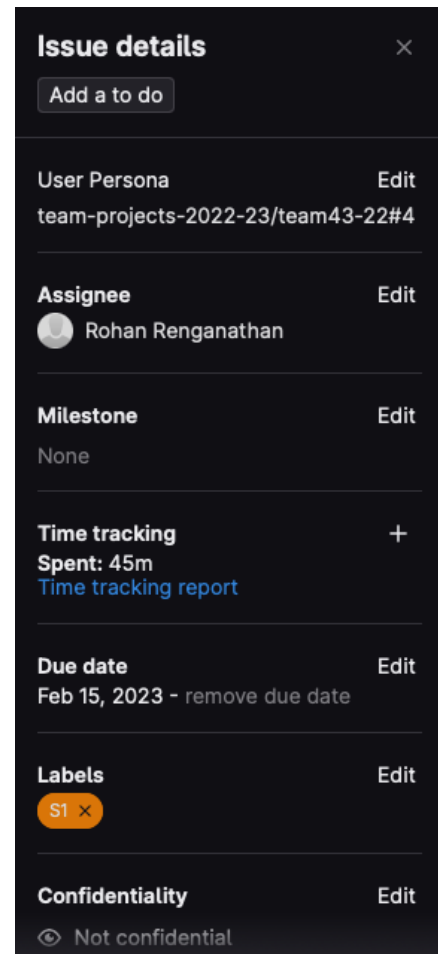


Figure 4

The above image displayed in *Figure 4* is the second Kanban card I created. It shows the user persona I created from what we believe to be an average user of the app would be. It is also labelled under S1, and this is part of Submission 1. This is also not confidential for the same reason as mentioned above.

The above 2 Kanban Cards are tasks that I have completed. I am still yet to add cards which are specific to the mockup I created.

For instance, this may include *Add Expense* which will allow the user to input the amount, date and category of an expense.

1. Design and implement the expense input form
2. Validate user input and display error messages if necessary
3. Save the expense to the database

Git Commit

Below is a copy of my terminal screen.

```
rohan@Rohans-MacBook-Air .ssh % git clone git@git.cs.bham.ac.uk:rxr105/helloworld.git
Cloning into 'helloworld'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (3/3), done.
rohan@Rohans-MacBook-Air .ssh % ls
helloworld  id_rsa  profile  id_rsa.pub  README  known_hosts  known_hosts.old
rohan@Rohans-MacBook-Air .ssh % cd helloworld
rohan@Rohans-MacBook-Air helloworld % ls
README.md
```

Figure 5

In *Figure 5*, I have created a repository called *helloworld* where I am now cloning the git repository into. At the end of this process the README.md file is present inside the repository.

```
rohan@Rohans-MacBook-Air helloworld % touch profile.txt
rohan@Rohans-MacBook-Air helloworld % start profile.txt
zsh: command not found: start
rohan@Rohans-MacBook-Air helloworld % open profile.txt
rohan@Rohans-MacBook-Air helloworld % ls
README.md  profile.txt
rohan@Rohans-MacBook-Air helloworld % git add profile.txt
rohan@Rohans-MacBook-Air helloworld % git commit -m "added name"
[main 6791b80] added name
  Committer: Rohan Renganathan <rohan@Rohans-MacBook-Air.local>
  Your name and email address were configured automatically based
  on your username and hostname. Please check that they are accurate.
  You can suppress this message by setting them explicitly. Run the
  following command and follow the instructions in your editor to edit
  your configuration file:

    git config --global --edit

  After doing this, you may fix the identity used for this commit with:

    git commit --amend --reset-author

  1 file changed, 1 insertion(+)
  create mode 100644 profile.txt
```

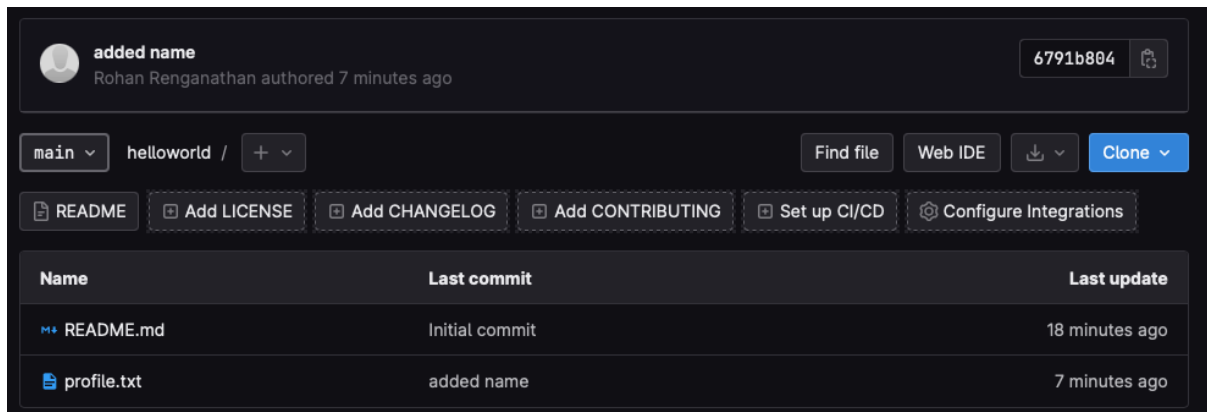
Figure 6

In the above screenshot, I am creating a new file called *profile.txt*. After typing the name of the persona into this file, I save and close it. Next, I add this file the git repository by committing it.

```
rohan@Rohans-MacBook-Air helloworld % git push
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 4 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 301 bytes | 301.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
To git.cs.bham.ac.uk:rxr105/helloworld.git
76ff326..6791b80  main -> main
rohan@Rohans-MacBook-Air helloworld % git pull
Already up to date.
```

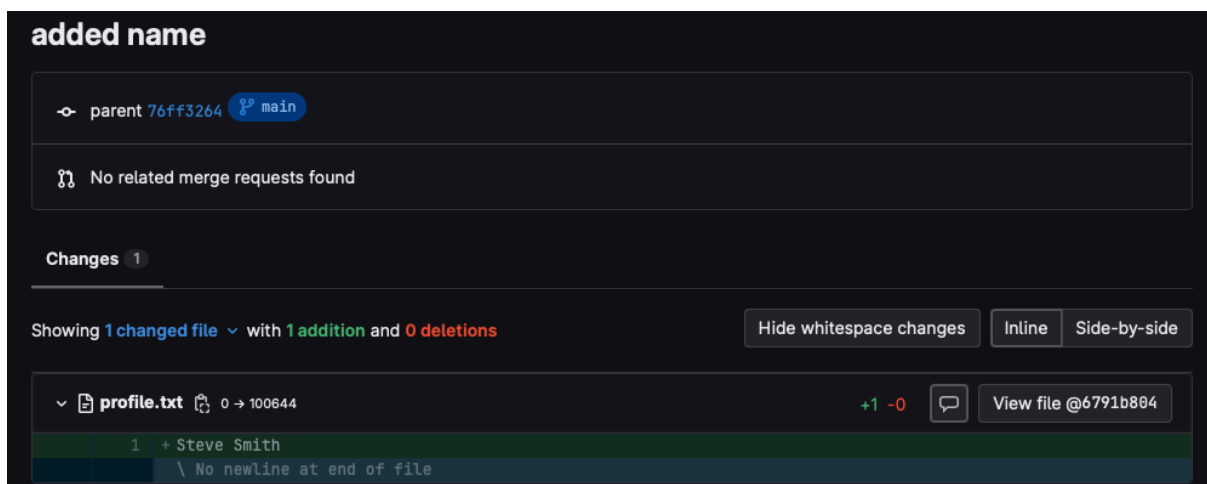
Figure 7

I then run the final command as displayed in *Figure 7* which is the git push command. This will confirm the changes I have made and push it to the git repository.



The screenshot shows the GitHub repository page for 'helloworld'. At the top, a commit titled 'added name' by Rohan Renganathan is shown, with the commit hash 6791b804. Below this, there are buttons for 'Find file', 'Web IDE', 'Clone', and 'Add'. A table lists the files in the repository:

Name	Last commit	Last update
README.md	Initial commit	18 minutes ago
profile.txt	added name	7 minutes ago



The screenshot shows the details of the commit 'added name'. It displays the parent commit 76ff3264 and the current commit 6791b804. Below this, it shows the changes made in this commit:

Showing 1 changed file with 1 addition and 0 deletions

The file 'profile.txt' is shown with the following content:

```
1 + Steve Smith
\ No newline at end of file
```

Figure 8

The above 2 images represented in *Figure 8* shows the changes have been committed successfully. From the image it says the new file, *profile.txt* is now present with the contents *Steve Smith* which is the name of the persona.