



**Dorset College Dublin**

# **SmartWage**

## **Final Year Project Report**

### **Bachelor of Science in Computing**

#### **BSC30924**

Filipe Lutz Mariano 25956

Vinicius Miranda 70973

Supervisor: Wenhao Fu

20/04/2025

## Abstract

This final year project encompasses the entire development lifecycle of SmartWage, from conceptualisation to deployment. The application is built using Kotlin for Android development, leveraging Firebase Authentication for secure user management, Firestore for cloud data storage and Room database to store user data locally.

The project follows the Agile software development methodology, allowing for iterative improvements based on continuous feedback. A detailed project plan outlines weekly milestones, beginning with research and requirement gathering, followed by UI/UX design, system architecture development, implementation of core functionalities, and rigorous testing. Furthermore, users can view their tax breakdown, save income and expenses and check if they are eligible for a tax refund.

A significant portion of this report focuses on demonstrating technical proficiency through system architecture design and database optimisation. Multiple images, diagrams, and charts will be used to illustrate the application's workflow, data processing mechanisms, and security implementations. Performance considerations, scalability challenges, and potential improvements will also be discussed.

The goal of this project is to enhance financial awareness among part-time workers, empowering them to make informed decisions regarding taxation and personal finance. By providing an accessible, user-friendly tool, SmartWage has the potential to bridge the gap between complex tax regulations and the everyday needs of students and employees. This report documents the research, development, and implementation processes in detail, ensuring a comprehensive demonstration of technical and analytical capabilities required for an advanced software engineering project.

## Declaration

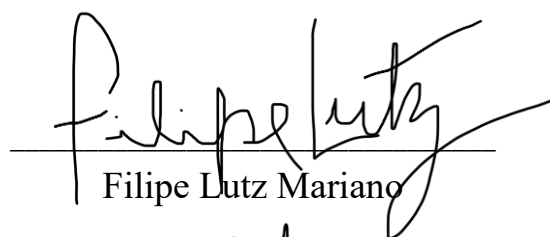
We are aware of the Dorset College policy on plagiarism and certify that this thesis is our own work.

The incorporation of material without formal and proper acknowledgement (even with no deliberate intent to cheat) can constitute plagiarism. If we have received significant help with a solution from one or more classmates, we will document this in my submitted work. If we have any doubt as to what level of discussion or collaboration is acceptable, we will consult our lecturer.


We understand that plagiarism is considered an act of fraudulence and an offence against the Dorset College Quality Assurance Manual. Alleged plagiarism will be investigated and dealt with appropriately by the Institute. We are also aware of the penalties outlined in the Institute Handbook.

Collusion is defined as more than one person working on an individual assessment. This includes jointly developed solutions as well as one individual giving a solution to another who then makes modifications and submits it as their own work.

We confirm that we have taken appropriate measures to protect our work from being copied or accessed by others. This includes safeguarding digital copies and not leaving printed materials where they can be duplicated. We acknowledge that failure to protect our work could bring us under suspicion of academic misconduct.



Filipe Lutz Mariano



Vinicius Miranda

20/04/2025

## Acknowledgements

We would like to sincerely thank our final year project supervisor, Wenhao Fu, for his continuous support and guidance throughout the development of this project from start to finish. His invaluable feedback and encouragement have played a crucial role in the completion of this project.

We would also like to extend our gratitude to all the test users who participated in the testing phases. Their thoughtful feedback and constructive criticism were instrumental in refining the features and ensuring the project met its objectives.

Finally, we would like to express our heartfelt thanks to our families, friends, and classmates who offered unwavering support throughout this project. Their patience, motivation, and encouragement made this challenging yet rewarding journey possible.

# Table of Contents

<b>Abstract .....</b>	<b>2</b>
<b>Declaration.....</b>	<b>3</b>
<b>Acknowledgements .....</b>	<b>4</b>
<b>1. Introduction.....</b>	<b>7</b>
1.1 Theme.....	7
1.2 Problem Statement.....	7
1.3 Hypothesis.....	8
1.4 Objectives .....	8
1.4.1 Main Objectives .....	8
1.4.2 Specific Objectives .....	8
<b>2. Research .....</b>	<b>9</b>
2.1 Related Research .....	9
2.2 Competitive Landscape.....	9
2.3 User Requirements.....	10
2.3.1 Personas .....	10
2.3.2 User Stories .....	11
2.4 Platform .....	11
<b>3. System Design .....</b>	<b>12</b>
3.1 Layout and Colours.....	12
3.2 Software Development Methodology .....	13
3.3 Use Case & Diagrams .....	14
3.4 Wireframes and Design Evolution .....	16
3.5 Software Architecture.....	17
3.6 Technical Architecture .....	18
3.7 Security Considerations .....	18
3.8 Summary.....	18

<b>4. Implementation</b>	19
4.1 Source Code Overview	19
4.1.1 Login and Authentication Module	19
4.1.2 Income Tracking Module	20
4.1.3 Expense Tracking Module	20
4.1.4 Tax Calculation Module	20
4.1.5 Dashboard Screen	21
4.1.6 Settings and Profile Management	21
4.2 API and Database Integration	22
4.3 Challenges and Solutions	22
<b>5. System Validation</b>	23
5.1 Usability Testing	23
5.2 Unit Testing	24
<b>6. Future Features and Improvements</b>	25
6.1 Features	25
6.2 Security and Privacy Enhancements	25
6.3 Accessibility and Interface	25
<b>7. Conclusion</b>	26
<b>8. Prototype</b>	27
<b>9. References</b>	31
<b>10. Appendix</b>	33
10.1 Appendix A - User Stories	33
10.2 Appendix B - Survey Questions	38
10.3 Appendix C - Survey Suggestion	44
10.4 Appendix D - Survey Frustration	45
10.4 Appendix E - Backlog GitHub Projects	46
10.4.1 Appendix E - Backlog Summary	46

# 1. Introduction

Taxation and financial management remain complex and challenging for part-time workers, especially students balancing multiple jobs and academic commitments. The SmartWage mobile application simplifies tax calculations, expense tracking, and refund estimation for hourly wage earners in Ireland. The project is motivated by firsthand experiences of difficulties in understanding tax liabilities, emergency tax deductions, and available tax reliefs.

Many part-time workers struggle to reconcile their tax positions, often resorting to costly professional assistance. SmartWage addresses these issues by offering an intuitive platform that allows users to input their income and expenses, estimate tax obligations, and identify potential refunds.

“It is important to make sure that taxes are handled right from the start and that the new employer deducts the correct amount of tax from your pay” (Citizens Information, 2023).

## 1.1 Theme

This project theme aims to improve financial knowledge and simplify tax processing requirements for part-time workers in Ireland, among students and those receiving hourly wages. The population faces major financial stress because their income levels change frequently, and they have limited knowledge about their tax obligations and complicated tax relief rules. The platform SmartWage assists user challenges by presenting an accessible system which enables users to handle their finances efficiently.

## 1.2 Problem Statement

Workers in Ireland who hold part-time positions struggle with comprehension of their tax requirements and handling of expenses as well as obtaining valid tax refunds or relief benefits. Complexity in their tax situation forces workers to use professional tax services, which makes their financial situation worse and adds unnecessary expenses. The project highlights the necessity to develop a user-friendly system which delivers both complete financial transparency and simple tax calculation. Hundreds of millions of euros in tax rebates go unclaimed in Ireland every year (Conor Pope, 2025).

## 1.3 Hypothesis

The motive to develop a mobile app is to provide the easiest way to track incomes, expenses and tax liabilities, and at the same time, offer tax relief guidance. The users will have improved management of their finances, less stress and the ability to claim tax refunds correctly.

## 1.4 Objectives

### 1.4.1 Main Objectives

The primary goal of this project is to create the SmartWage mobile application that makes part-time workers simplify income tracking, expense control, tax calculation and refund estimation, which is available.

### 1.4.2 Specific Objectives

- Develop a user interface that will combine ease of use with financial literacy capabilities.
- Implement secure methods of user authentication and data storage systems.
- Offer precise tax computation services in real time, which follow Irish Revenue criteria.
- Users will be enabled to estimate potential refunds.
- Provide educational resources to enhance users' understanding of financial management and tax-related processes.



## 2. Research

The research was essential to establish a foundation for the development of the SmartWage mobile application. This section explores existing research, related projects, and competitive solutions that address similar issues. Understanding the competitive landscape, user requirements, and previous research provides insight into the best practices and innovations that can be incorporated into the development process.

### 2.1 Related Research

Existing literature highlights the financial struggles faced by part-time workers, particularly in managing fluctuating incomes and understanding tax liabilities. Research has demonstrated that financial literacy tools can significantly improve users' confidence in managing their finances. Mobile applications that provide real-time financial data, tax calculators, and educational resources have been shown to reduce stress and improve budgeting skills. Studies emphasise that user-friendly interfaces and secure data handling are crucial factors in ensuring the success of financial management applications.

### 2.2 Competitive Landscape

Several financial management applications exist in the market, including Mint, YNAB (You Need A Budget), and TaxAssist. However, many of these solutions are tailored for salaried employees or are limited in their ability to address local tax regulations in Ireland. While some apps offer general budgeting and tax estimation features, few provide targeted solutions for hourly wage earners with multiple income sources or specific guidance on claiming tax reliefs. SmartWage aims to fill this gap by providing a dedicated solution that complies with Irish tax regulations and focuses on part-time workers' unique financial needs.

## 2.3 User Requirements

Understanding user requirements is critical for developing an application that meets the needs of its target audience. Primary user requirements for SmartWage include:

- Income and expense tracking: Users should be able to enter and save multiple income and expenses.
- Tax estimation: Tax calculations based on Irish tax regulations.
- Refund eligibility: The ability to identify potential tax refunds and provide guidance on how to claim them.
- Secure data storage: Ensuring the safety and privacy of user data.
- User-friendly design: An intuitive interface that simplifies financial management tasks.

### 2.3.1 Personas

To ensure the SmartWage application meets the needs of its diverse user base, we have identified a range of personas that reflect the primary target audience. The app is designed to benefit anyone working part-time without a fixed salary, helping them to manage finances, track income, and estimate tax obligations efficiently.

1. Student: A university student balancing coursework and multiple part-time jobs. Needs to track income fluctuations and claim educational tax reliefs.
2. Retail Worker: Works shifts with irregular pay. Wants a simple way to log income and receive alerts about tax liabilities.
3. Hospitality Staff: Works in restaurants, bars, or hotels, earning a combination of hourly wages and tips. Needs to calculate overall taxable income effectively.
4. Seasonal Employee: Works on a short-term basis, such as holiday retail or agricultural jobs. Needs to track limited-time earnings and potential refund eligibility.
5. Part-Time Contractor: Engaged in temporary or project-based work. Requires insights on tax calculations and deductions applicable to contract work.
6. Carer or Nanny: Works part-time in caregiving roles with variable income and needs tax estimates for reporting earnings.
7. Construction Worker: Works on an hourly basis with fluctuating workloads. Needs to manage multiple short-term contracts and tax obligations.
8. Event Staff: Works in entertainment or corporate events on an irregular schedule. Requires tools to log multiple income sources and estimate tax liability.

9. Warehouse or Factory Worker: Works shift-based hours that change weekly. Needs financial planning tools to accommodate varying pay periods.
10. Ride-Share Driver: Earns income through platforms like Uber or Bolt, requiring tax estimation and expense tracking for deductions.

### 2.3.2 User Stories

User stories serve as fundamental components for defining essential features in the SmartWage application, so it addresses the requirements of financial managers who are students or part-time workers. The system contains distinct stories which document essential user needs, including the tracking of income and taxes alongside the prediction of refunds and expense tracking capabilities. Each user story maintains a standardised format which contains both priority designations as High, Medium or Low alongside separate acceptance criteria that verify proper system functionality. Account creation functions, income tracking abilities and data protection measures are among the high-priority stories of the application, while medium and low-priority stories introduce features that improve user experience through notifications, tax breakdown explanations and customisation capabilities. The complete list of 20 user stories with their priority and acceptance criteria is presented in *(10.1 - Appendix A – User Stories)*.

## 2.4 Platform

One of the initial stages of the project was brainstorming and determining the project parameters, specifically the platform for the application development, either Android Studio for a mobile app or a web application. Both platforms are suitable for this project. However, while evaluating the concept, the development of a mobile application appeared more appropriate.

Studies have shown that mobile apps are more popular than websites, as they are more user-friendly. Mobile apps cater to better user experience, load content faster, and are very easy to use. Apps provide users the ability to perform a task better with the help of gestures. It is easier to navigate through, unlike the website (Whiz Solutions, 2019).

## 3. System Design

SmartWage was created with simplicity and transparency as its core principles. The target audience includes part-time workers and students, many of whom have insufficient financial expertise. Consequently, the application must be user-friendly, aesthetically pleasing, and clear of complicated financial language. This chapter explores the principal design choices, including layout, colour palettes, technique, and architectural framework that directed the app's development.

### 3.1 Layout and Colours

The design uses a modular card-based format, enabling users to instantly identify functionalities such as adding income, expense entry, and tax breakdown. Buttons are properly positioned and colour-coded, with feedback delivered through dialogue boxes and discreet warnings. The colour scheme was inspired by the Digital Synopsis design resource, highlighting accessibility and readability.

- Primary Colour: Soft Blue – designated for buttons and accents.
- Secondary Colour: Neutral Grey – designated for background and text.
- Accent Colour: Light Green - indicative of good feedback (such as tax refunds).
- Red highlights: For notifications or tax obligation reminders.

The user interface design was initially prototyped in Figma and subsequently developed with Jetpack Compose components in Android Studio. Icons were maintained in a minimalistic and uniform manner throughout the screens.

## 3.2 Software Development Methodology

The Agile methodology was chosen due to its flexibility, iterative structure, and feedback-oriented development process. The team split work into sprints, each concentrating on a certain set of features based on user stories.

Principal agile methodologies employed:

- Weekly Zoom meetings with supervisor feedback.
- Continuous testing after each feature deployment.
- Regular commits on GitHub for version control.
- GitHub Projects for task management and priority designation.

Each sprint started with a planning session, succeeded by feature development, integration, testing, and the inclusion of feedback. At the conclusion of the development phase, all high-priority user stories were resolved.

“Agile is a type of software development methodology that anticipates the need for flexibility and applies a level of pragmatism to the delivery of the finished product. Agile software development requires a cultural shift in many companies because it focuses on the clean delivery of individual pieces or parts of the software and not on the entire application” (Robinson S., 2024).

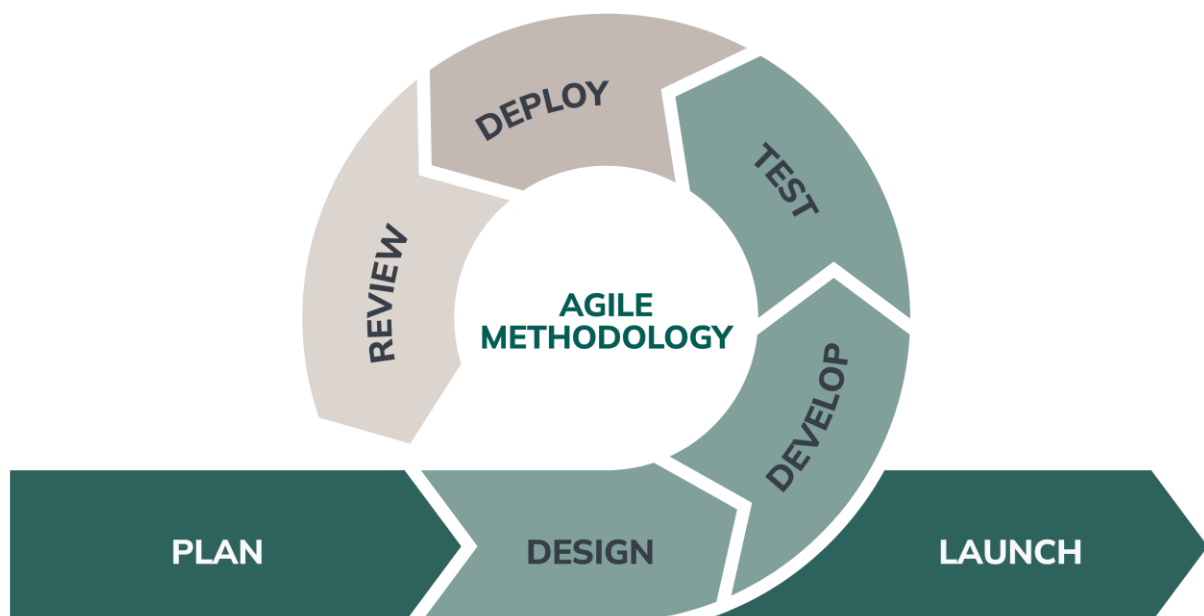


Image 3.2.1 - Diagram of Agile Methodology

### 3.3 Use Case & Diagrams

A use case diagram was developed to illustrate the application's major functions. It shows the interaction between the user and the SmartWage system:

Actor:

- User

Use Cases:

- Login
- Sign up
- Add income
- Add expenses
- View tax breakdown
- Estimate refunds
- Access tutorials
- Update profile
- Update settings

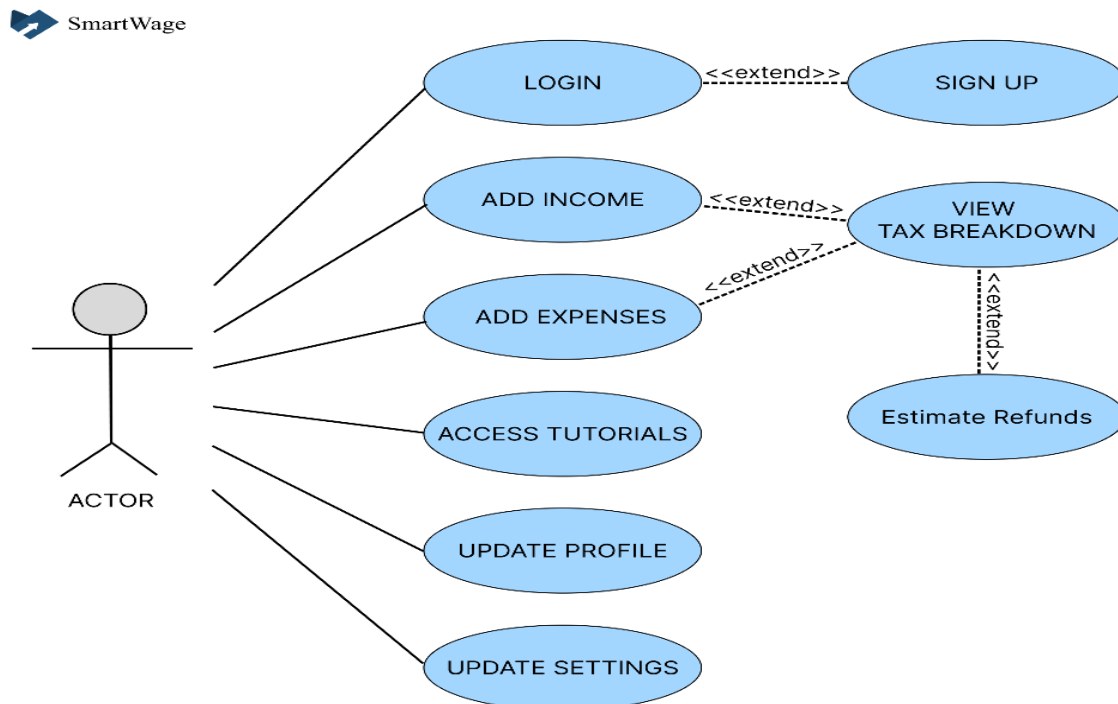


Image 3.3.1 - Use Case Diagram SmartWage

UML Class Diagrams and Activity Diagrams were used to better understand the project flow and connectivity between classes and methods.

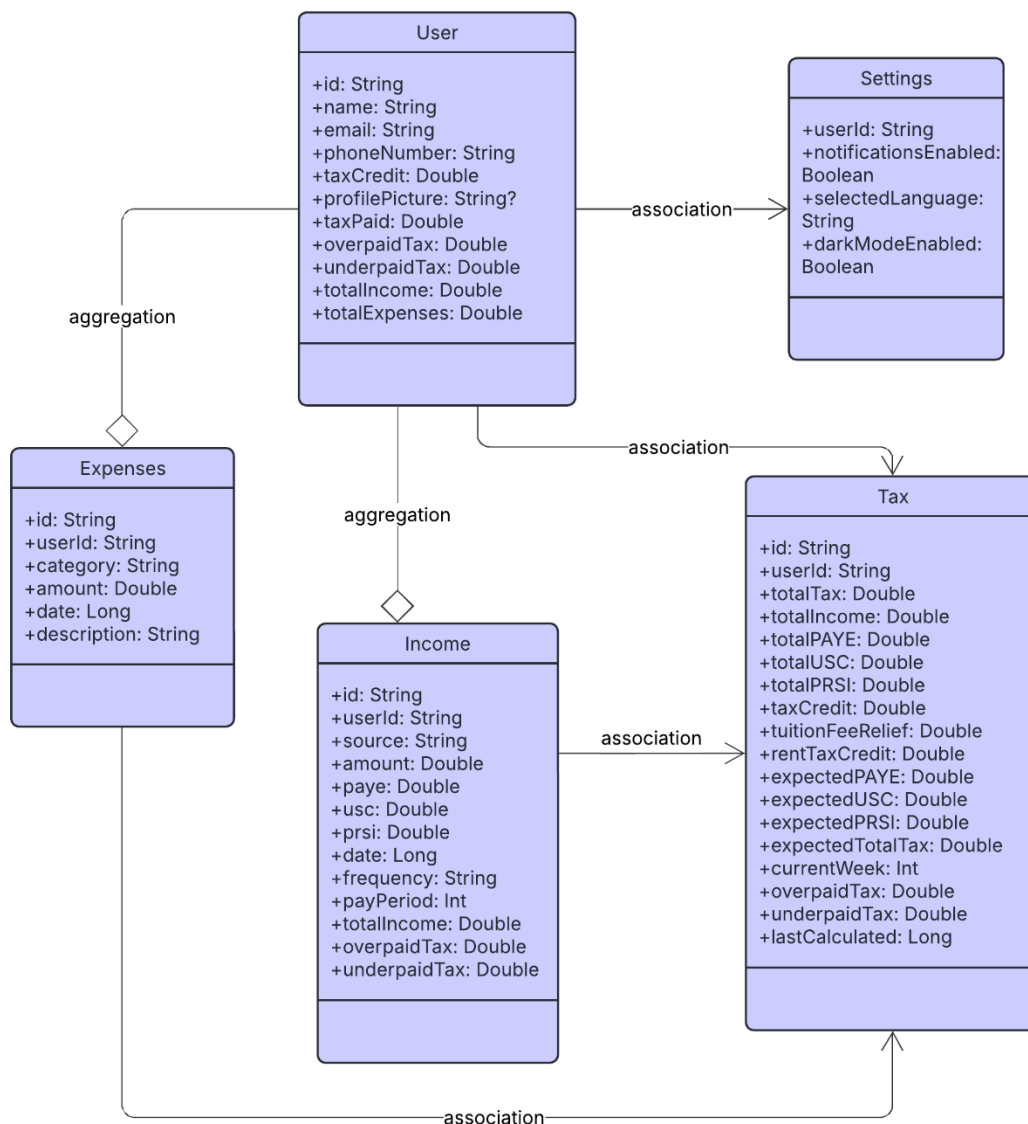


Image 3.3.2 - UML Class Diagram

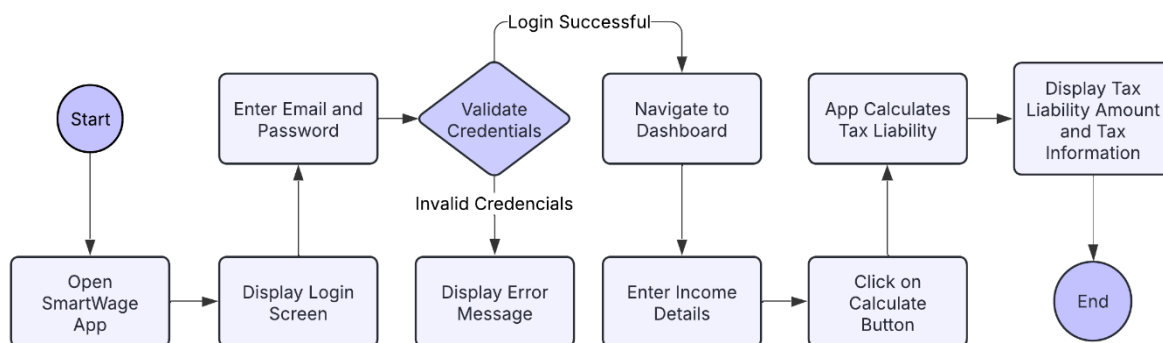


Image 3.3.3 - Activity Diagram - Quick Tax Calculation

### 3.4 Wireframes and Design Evolution

At the initial phases of the SmartWage project, wireframes were created to illustrate the user interface architecture and feature progression prior to the commencement of execution. These included low-fidelity and high-fidelity digital prototypes designed using Figma.

The low-fidelity wireframes were used as a layout prototype to outline key screens and components, such as the login, dashboard, income and expense tracking, and settings sections. The original proposal contained supplementary features such as adding job information and login choices through Google, Facebook, and Apple. However, the project scope and technical constraints led to the removal of these features in later stages.

As development progressed, the design became an improved, user-friendly interface that emphasised intuitive income and tax tracking. High-fidelity wireframes improved layout aesthetics, interactions, and usability, directing frontend execution.

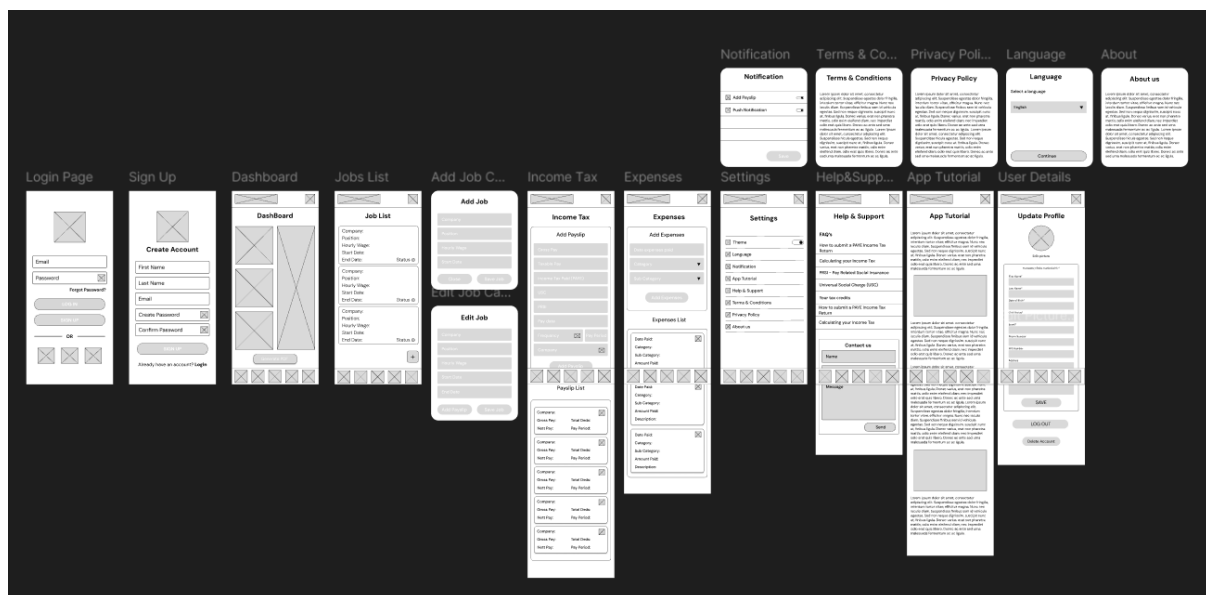


Image 3.4.1 - Low-fidelity Wireframes



High-fidelity design that was closest to the final product version.

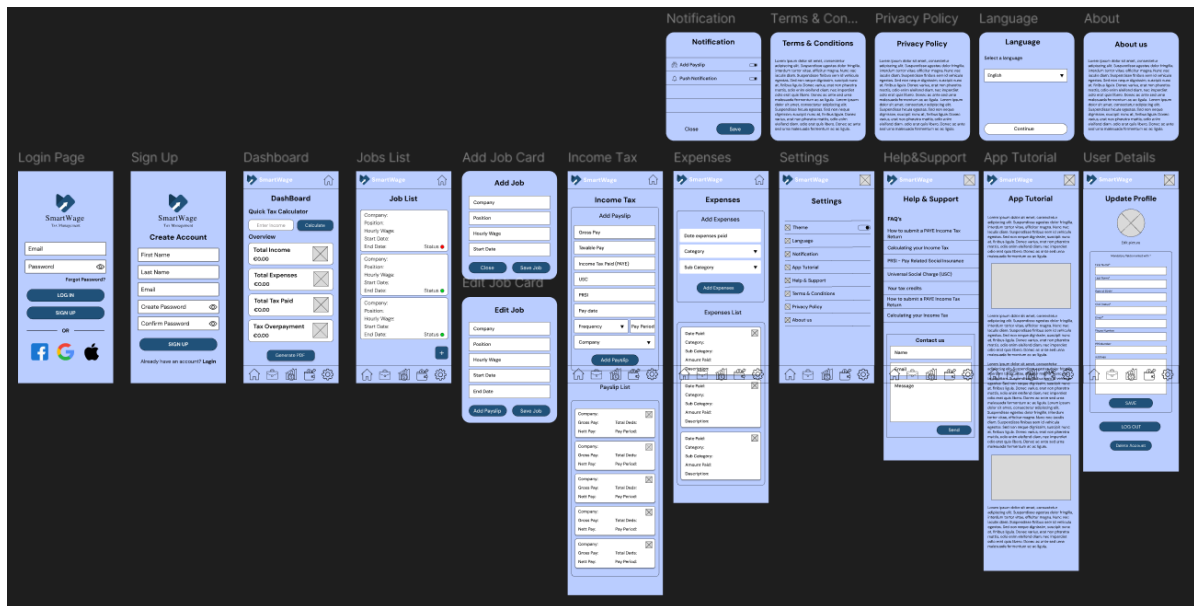


Image 3.4.2 - High-fidelity Wireframes

### 3.5 Software Architecture

The SmartWage architecture follows the MVVM pattern (Model-View-ViewModel) to establish roles while improving modularity.

- **Model:** Data classes for user, income, expenses and tax.
- **View:** Composable functions using Jetpack Compose that display live data and user interactions.
- **ViewModel:** Handles state and logic, exposing StateFlow to the UI.

Key components:

- AuthViewModel, IncomeViewModel, ExpenseViewModel, TaxViewModel, DashboardViewModel and SettingsViewModel.
- All data are stored in Room (local Database) and Firebase Firestore (cloud Database).

As Android apps grow, it's important to define an architecture that allows the app to scale, increases the app's robustness, and makes it easier to test (Guide to app architecture, 2025).

## 3.6 Technical Architecture

SmartWage tech stack and data flow:

- Language: Kotlin in Android Studio
- Frontend: Jetpack Compose for UI
- Backend: Firebase Authentication, Firestore for cloud storage, Room as a local database.
- Security: Firebase rules, Sign up confirmation by email before login.
- APIs: Tax calculation logic coded manually from 2025 Revenue.ie guidelines.

SmartWage uses dependency injection via Hilt and structured navigation via the Navigation component with Destinations.kt and AppNavigation.kt.

Hilt is a dependency injection library for Android that reduces the boilerplate of doing manual dependency injection in your project (Dependency injection with Hilt, 2025).

## 3.7 Security Considerations

User financial data is extremely sensitive. SmartWage implements the following:

- Encrypted local storage using Room.
- Firebase Authentication with email and password login.
- Optional password reset and account deletion flows.
- Firestore security rules to ensure users can only access their information.
- Secure password validation (min. 8 characters, uppercase, digit, special character).

Future improvements (see Chapter 6) include adding two-factor authentication.

## 3.8 Summary

SmartWage's design emphasises simplicity, modular architecture, and data security. The team developed the app from concept to prototype, ensuring it was based on genuine user requirements, followed best practices in Android development, and was supported by solid financial logic. Each design choice, including the colour palette and navigation structure, contributes to creating a simple experience for part-time employees managing complex tax information.

## 4. Implementation

The development of SmartWage required transforming comprehensive design and planning into a fully working Android application. This chapter includes a detailed analysis of the system modules, essential features, applied technology, and obstacles faced during development. The application was developed using Kotlin and Jetpack Compose in Android Studio, supported by Firebase for its backend infrastructure. The application includes a comprehensive offline mode with Room database caching and Hilt for dependency injection.

### 4.1 Source Code Overview

SmartWage was developed in a modular structure, separating key responsibilities across packages:

- ViewModels: Handle logic and business state.
- Repositories: Provide data bridges between Firestore, Room and ViewModels.
- Data Layer: Contains local database access objects (DAOs) and entity models.
- Navigation: Defined in Destinations and AppNavigation functions for clear screen routing.
- Utils: For tax logic with the TaxCalculator function, validation rules and constants.

Hilt facilitates dependency injection across modules, improving reliability and code reuse. All code is thoroughly documented with inline comments and exception logging implemented via Timber. “If you want your code to be easy to write, make it easy to read” (Martin, 2015).

#### 4.1.1 Login and Authentication Module

The login screen, developed with Jetpack Compose, enables users to authenticate via email and password, reset forgotten passwords, and register for new accounts.

Key functionalities:

- Firebase Authentication handles login and signup actions.
- Email validation and password rules are enforced via the ValidationUtils function.
- User information is synchronised with Firestore upon successful authentication.

The AuthViewModel.kt handles all login, signup and reset operations and uses AuthRepository.kt to interact with Firebase securely.

### 4.1.2 Income Tracking Module

This module enables users to enter income. The income fields are:

- Employer (Company) name.
- Income date.
- Frequency (Weekly, Fortnightly and Monthly).
- Gross income amount.
- Taxes (PAYE, PRSI and USC).

The income information is stored locally and in the cloud. Offline support allows users to access prior entries without an internet connection.

File: IncomeViewModel.kt

Repository: IncomeRepository.kt

The system uses FirebaseAuth to ensure that the user can only see their own data.

### 4.1.3 Expense Tracking Module

Users can add expenses categorised as:

- Rent Amount (Rent Tax Credit)
- Tuition Fee Amount (Tuition Fee Relief)

Each category contains amount paid, expense date and description that is optional. It is in the tax credit calculations.

File: ExpenseViewModel.kt

Storage: Firestore and Room

Logic support: TaxCalculator.kt.

### 4.1.4 Tax Calculation Module

This is SmarWage's core feature. Based on saved income and expenses, the app calculates:

- Taxes (PAYE, PRSI, USC).
- Tuition Fee Relief if applicable (20% beyond the first €3,000 up to €7,000).
- Rent Tax Credit if applicable (20% of rent amount paid up to €1,000).
- Refund eligibility with overpaid or underpaid tax estimation.

The calculations are specified in `TaxCalculator.kt`, including functions as `calculateTax`, `calculatePAYE`, `calculatePRSI`, `calculateUSC`, `calculateTuitionFeeRelief` and `calculateRentTaxCredit`.

ViewModel: `TaxViewModel.kt`.

Output:  $\text{Tax paid} - \text{Expected tax} = \text{Tax difference} = \text{Overpayment or Underpayment tax}$ .

UI: Tax breakdown in `TaxCreditScreen.kt`.

#### 4.1.5 Dashboard Screen

The dashboard includes:

- Top scaffold bar with SmartWage logo and user icon or photo, if any.
- Quick tax calculation field and button.
- Overview of total income, expenses, tax paid, and overpaid or underpaid tax, if any.
- Bottom scaffold bar with navigation icons and texts.

`DashboardViewModel.kt` combines data from income, expenses, and tax calculations to provide an accurate overview. The screen refreshes automatically upon the addition of new data.

#### 4.1.6 Settings and Profile Management

The app allows users to:

- Update personal information such as name and phone number.
- Add, edit or delete a profile photo.
- Enable or disable notifications.
- Send a link to the user email to update the password.
- Search for tutorial videos to see how to use the app.
- Log out or delete the account permanently.

Implemented with `ProfileViewModel.kt` and `SettingsViewModel.kt`. Firebase Storage facilitates profile photo uploads, while Firestore and Room maintain synchronised user data.

## 4.2 API and Database Integration

SmartWage runs with:

- Firebase Authentication: For secure user signup and login.
- Firestore (Cloud): To store income, expenses, tax and user profile information.
- Room (Local Database): Offline storage of data and faster access.
- Jetpack Navigation: Smooth screen transitions.

Tax logic is coded manually using Revenue.ie guidelines, avoiding the need of external tax APIs.

## 4.3 Challenges and Solutions

Firebase Authentication Issues

- Initially, Firebase failed to synchronise user information with Room.
- Solution: Developed a two-way synchronisation in UserRepository.kt.

Income Tax Calculation Complexity

- Addressing frequency changes (weekly, fortnightly, monthly) involved the development of a multiplier system.
- Solution: Abstracted it using an Enum class within TaxCalculator.kt.

Security and Validation

- Ensure data privacy and secure access.
- Solution: Password rules, Firestore security protocols and local encryption measures.

Offline Access Reliability

- Users required offline access despite synchronisation failures.
- Solution: Room storage has been prioritised, and updates have been queued for when the network is restored.

Camera Authorisation for Profile Photo

- Experienced difficulties in enabling camera access for profile photo uploads due to missing authorisation declarations.
- Solution: Included required camera permissions into AndroidManifest.xml and handled runtime permission requests within the user interface layer.

The full project code can be found on the GitHub repository:

<https://github.com/FilipeLutz/SmartWage>.

## 5. System Validation

### 5.1 Usability Testing

Usability testing is crucial to verify that the application's functionality meets user expectations and usage requirements. In-person usability testing sessions for SmartWage were done with a group of about ten students, predominantly including part-time employees, reflecting the app's target audience.

Everyone who participated was instructed on fundamental use cases, including adding income and expenses, tax breakdown, and refund eligibility measurements. The developers monitored interactions and gathered feedback directly, facilitating efficient clarification and enhanced understanding of user behaviour.

The objectives of the testing included:

- Understanding navigation simplicity.
- Assessing the clarity of tax-related terminology.
- Evaluating the intuitiveness of data entry procedures.
- Measuring user comprehension of tax refund eligibility.

Participants were asked to express their thoughts while interacting with the app.

Key findings:

- About 90% of users found the app layout clear and easy to use.
- Several users proposed the inclusion of dialogue messages for complex tax definitions, like PAYE, PRSI and USC.
- Users that used different mobile screen sizes noted that spacing on screens could change from one screen size to another.

Improvements implemented based on feedback:

- A tax rate information card detailing PAYE, PRSI, and USC, with clickable links to the corresponding Revenue.ie website for further information on each tax calculation.

The practical nature of these sessions produced essential qualitative data, ensuring that SmartWage's design decisions were based on actual user experience rather than theory.

The primary purpose of usability testing is to uncover design flaws, identify areas for improvement, and find opportunities to enhance the user experience (The complete guide to usability testing, 2025).

## 5.2 Unit Testing

Unit testing was conducted across all main modules to guarantee feature stability and accuracy.

Testing focus:

- Tax calculation accuracy (TaxCalculator.kt).
- Validation utilities (email and password).
- Income and expenses entry logic.
- State flow updates in ViewModels.

Using JUnit and simulated data, it evaluated diverse income levels and deduction combinations, confirming that outputs aligned with Irish Revenue tax regulations for the 2024–2025 period.

Key Test Scenarios:

- PAYE, PRSI and USC calculations with different income frequencies.
- Tuition Fee Relief application thresholds (no tuition fee relief under €3,000).
- Overpayment tax refund estimate logic.
- Edge scenarios such as zero income or absent fields.

All tests were executed successfully, confirming that the business logic operated as intended and demonstrated resilience to input variations.

“Unit testing involves isolating individual components of a software system and verifying their correct behaviour. These components, often referred to as "units", could be functions, methods, or classes. The primary goal is to ensure that each unit performs its intended task accurately and reliably.” (HyperTest, 2025).



## 6. Future Features and Improvements

SmartWage was developed implementing a Minimum Viable Product (MVP) approach which concentrated on solving the immediate need of part-time workers to understand and deal with their Irish tax obligations. The future development plan adds updates and upgrades for enhancement of functionality, accessibility and scalability that will be integrated in the future versions. According to Atlassian, “The minimum viable product, or MVP, is the simplest version of a product that you need to build to sell it to a market” (Atlassian, n.d.).

### 6.1 Features

There is a list of future implementations for the project, such as to implement a PDF export feature for income and tax summaries and a payslip scanner using optical character recognition (OCR) to streamline data entry (Hasan, 2022). Also, notifications will notify users to stay consistent with financial log data, for example, by reminding them to upload weekly payslips. These features support a proactive behaviour and improve personal financial tracking (Lusardi, 2014).

### 6.2 Security and Privacy Enhancements

Firebase Authentication is used for a secure login, and Room database is used as local data storage. The plan is to introduce end-to-end encryption of sensitive data, as income and expense information should be available only to the authenticated user. In addition to the plan, user permission control will be added to enhance privacy and security. Furthermore, two-factor authentication is in the list of future developments to provide a more secure user login. These improvements follow suggested standards for secure mobile applications (Alqahtani, 2020).

### 6.3 Accessibility and Interface

Two features that were not complete were dark mode for low-light environments and multi-language support to accommodate Ireland’s multicultural workers. These features will be available in the next version. The plan is that with these additions SmartWage will become more inclusive and adaptable.

## 7. Conclusion

The development of the SmartWage mobile application combines technical, analytical, and creative skills into a real-world solution; the intention is to improve financial literacy and tax management among part-time workers in Ireland. Over the course of this project, it was learnt to apply essential software engineering practices, from UI/UX design to secure backend development, using Kotlin, Jetpack Compose, Firebase, and Room Database.

Working as a team, it followed an Agile approach and used tools such as GitHub and GitHub Project for collaboration and management, which guided the project to stay on track, manage tasks efficiently, and incorporate feedback quickly. It strengthened the problem-solving abilities through practical challenges, such as implementing accurate tax logic, handling multiple income frequencies, and ensuring secure data handling across both local and cloud databases.

One of the most challenging aspects was building a clear and responsive interface that could present complex tax information in a simple and user-friendly way. Adapting all the Revenue.ie guidelines into automated tax calculations required deep research and understanding, careful coding, and constant validation. Yet, overcoming this challenge became one of the most rewarding parts of the project.

The team is proud to have delivered a fully functional mobile application project that provides real-time tax estimations, tracks income and expenses, and offers refund eligibility, all features that can genuinely help users feel more in control of their finances. Seeing test users engage with the app and provide positive feedback reinforced the impact of the teamwork.

Overall, this experience has not only improved the team's technical skills but also shaped their confidence in building scalable and meaningful applications. The team believes SmartWage has strong potential for further development, and the knowledge gained through this project will be highly valuable in their future careers.

## 8. Prototype

The SmartWage application is straightforward and easy to navigate. This part contains screenshots of each interface.



Image 8.1 - Login

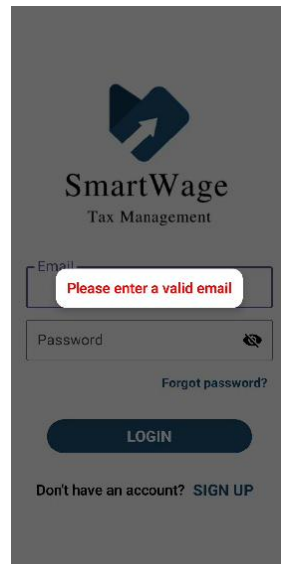


Image 8.2 - Email Error

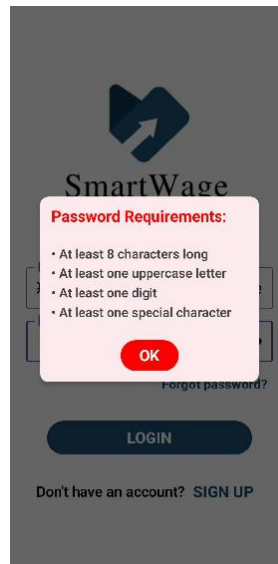


Image 8.3 - Password Error

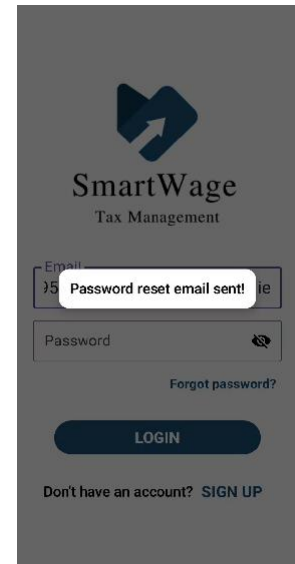


Image 8.4 - Password Reset



Image 8.5 - Sign Up

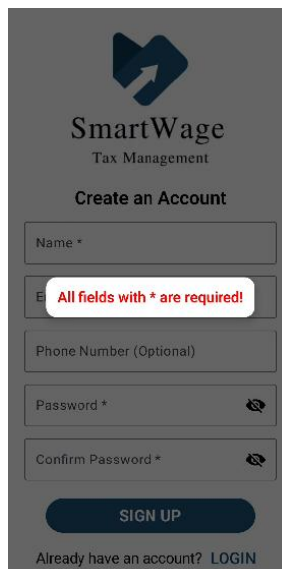


Image 8.6 - Error Handling

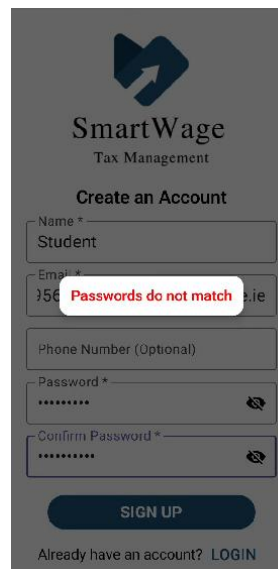


Image 8.7 - Password Error

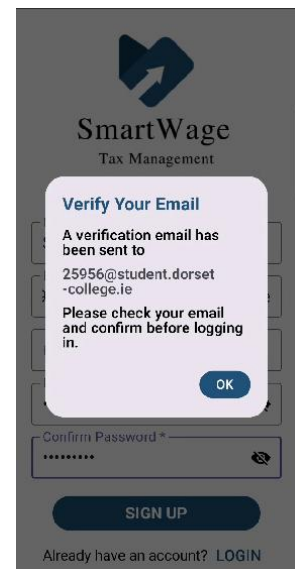


Image 8.8 - Verify Account

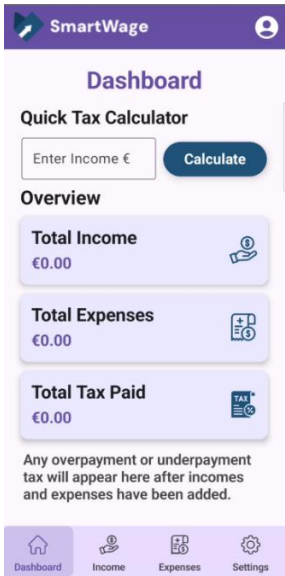


Image 8.9 - Dashboard

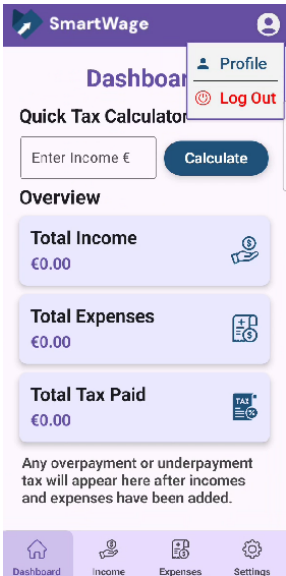


Image 8.10 - User box menu

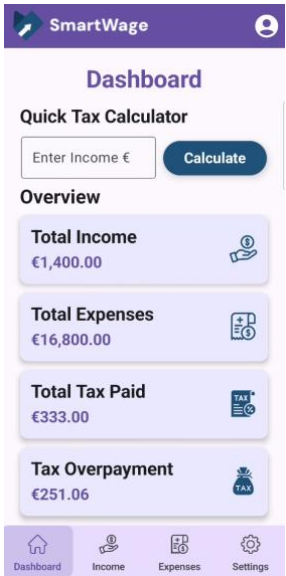


Image 8.11 - Overview data

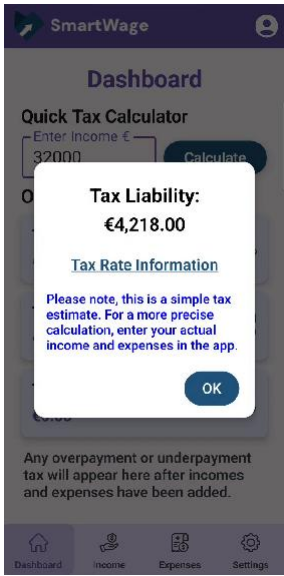


Image 8.12 - Tax Cal. Result

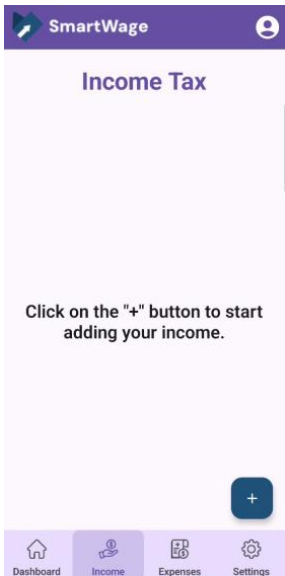


Image 8.13 - Empty Income

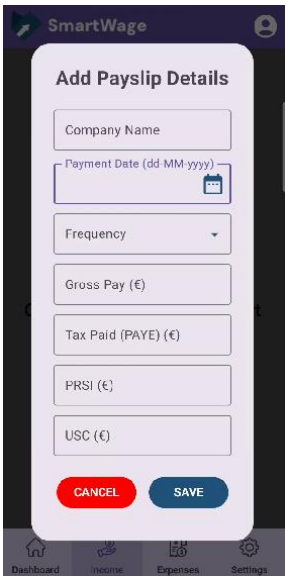


Image 8.14 - Add Income

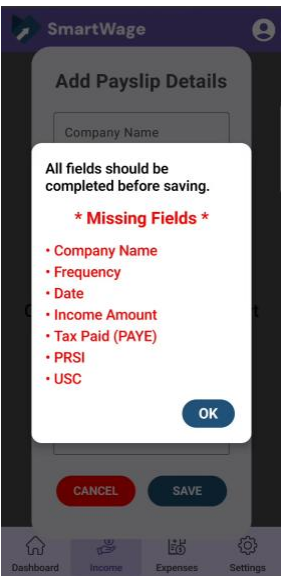


Image 8.15 - Fields Error



Image 8.16 - Income List

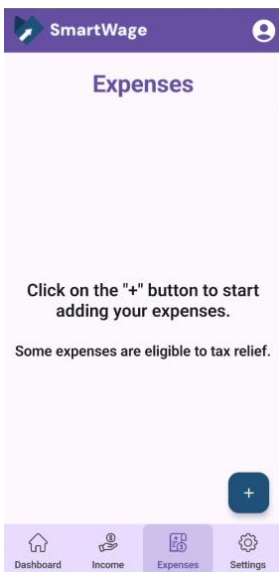


Image 8.17 - Expenses

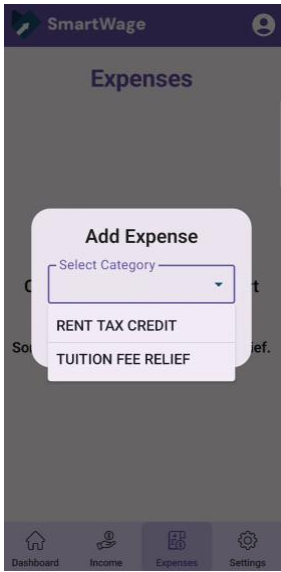


Image 8.18 - Add Expenses

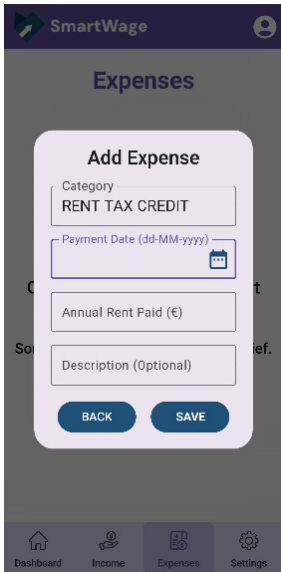


Image 8.19 - Add Rent

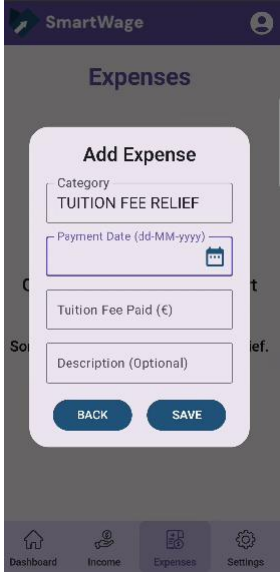


Image 8.20 - Tuition Fee



Image 8.21 - Expenses List

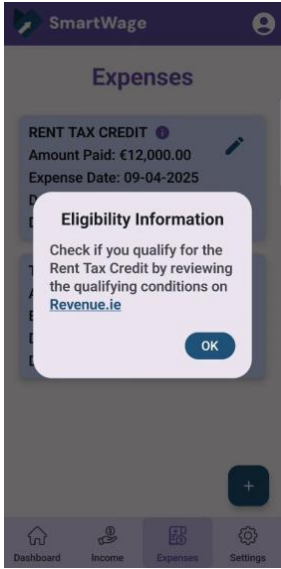


Image 8.22 - Rent Credit

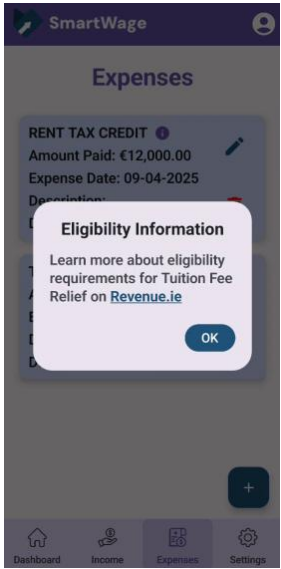


Image 8.23 - Tuition Relief

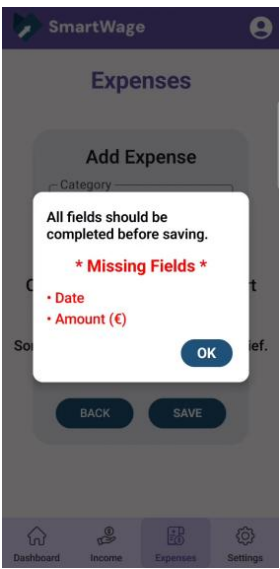


Image 8.24 - Error Dialog



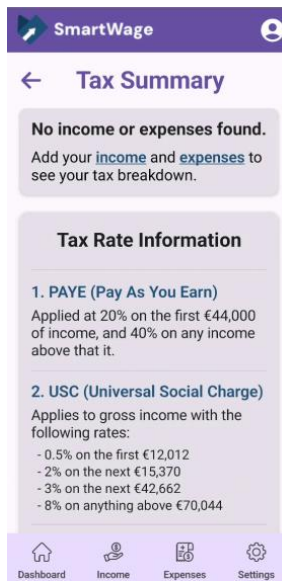


Image 8.25 - Tax Rate Info



Image 8.26 - Tax Summary

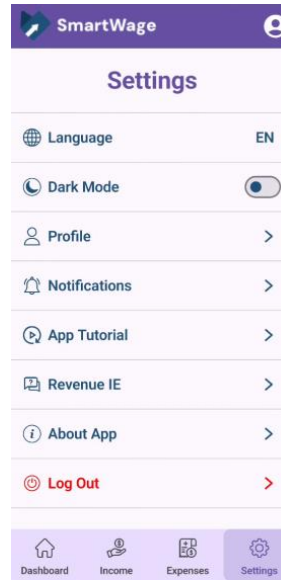


Image 8.27 - Settings

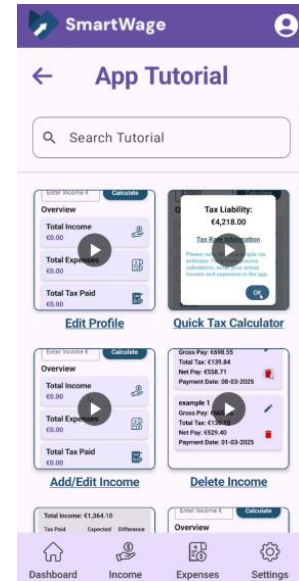


Image 8.28 - App Tutorial

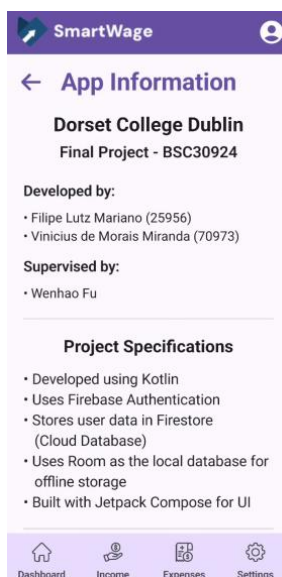


Image 8.29 - About App Info

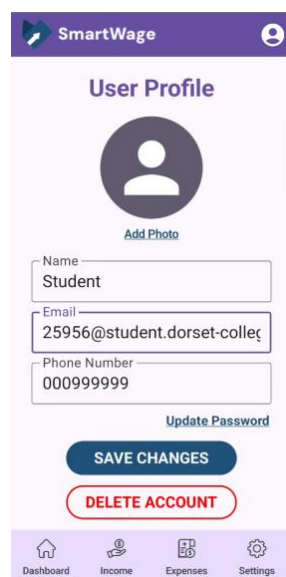


Image 8.30 - User Profile

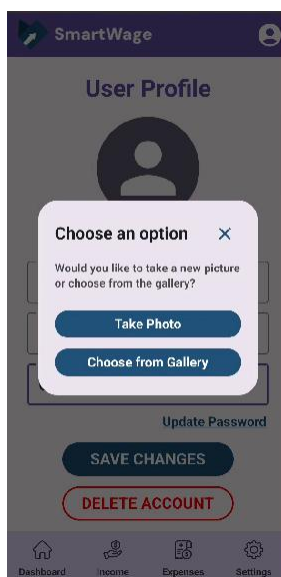


Image 8.31 - Add Photo

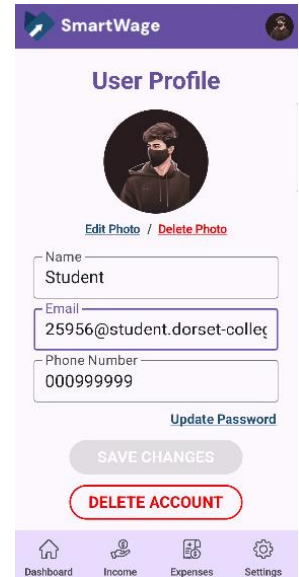


Image 8.32 - Edit Photo

## 9. References

- Alqahtani, A.K., 2020. *A survey of secure mobile cloud computing. Journal of Network and Computer Applications*, 138, pp.35–58.
- Citizens Information, 2023. *Tax and starting work* [online]. Citizens Information. Available at: <https://www.citizensinformation.ie/en/employment/starting-work-and-changing-job/starting-work/tax-and-starting-work/> [Accessed 16 Apr. 2025].
- Conor Pope and Nolan, A., 2025. *Tax refunds going unclaimed by up to 500,000 PAYE taxpayers – Revenue. The Irish Times*, [online] 4 Feb. Available at: <https://www.irishtimes.com/your-money/2025/02/04/tax-refunds-going-unclaimed-by-up-to-500000-payee-taxpayers-revenue/> [Accessed 16 Apr. 2025].
- Google, 2025. *Dependency injection with Hilt* [online]. Android Developers. Available at: <https://developer.android.com/training/dependency-injection/hilt-android> [Accessed 16 Apr. 2025].
- Google, 2025. *Guide to app architecture* [online]. Android Developers. Available at: <https://developer.android.com/topic/architecture#recommended-app-arch> [Accessed 16 Apr. 2025].
- Hasan, A.R., 2022. *OCR-based automated document recognition using mobile devices. International Journal of Computer Applications*, 184(27), pp.20–25.
- HyperTest, 2025. *Importance and purpose of unit testing in software engineering* [online]. Available at: <https://www.hypertest.co/unit-testing/importance-and-purpose-of-unit-testing-in-software-engineering#:~:text=Unit%20testing%20involves%20isolating%20individual,intended%20task%20accurately%20and%20reliably> [Accessed 16 Apr. 2025].
- Lusardi, A. and Mitchell, O.S., 2014. *The economic importance of financial literacy: Theory and evidence. Journal of Economic Literature*, 52(1), pp.5–44.
- Martin, R.C., 2015. *Clean Code: A Handbook of Agile Software Craftsmanship*. Upper Saddle River, NJ: Prentice Hall. Available at: <https://shorturl.at/kJYQH> [Accessed 16 Apr. 2025].

- Robinson, S. and Bolton, K., 2024. *Agile software development* [online]. TechTarget. Available at: <https://www.techtarget.com/searchsoftwarequality/definition/agile-software-development> [Accessed 16 Apr. 2025].
- Testing, U., 2025. *43 Beautiful Color Palettes For Your Next Design Project* [online]. Digital Synopsis. Available at: <https://digitalsynopsis.com/design/color-palettes-schemes-combinations/> [Accessed 16 Apr. 2025].
- UserTesting, 2025. *The complete guide to usability testing* [online]. Available at: <https://www.usertesting.com/resources/guides/usability-testing> [Accessed 16 Apr. 2025].
- Whiz Solutions, 2019. *7 reasons why mobile apps are better than websites* [online]. Available at: <https://www.whizsolutions.co.uk/7-reasons-mobile-apps-better-website/> [Accessed 16 Apr. 2025].
- Khan, S., n.d. *Introduction to Unit Testing in Android: Getting Started with the Basics – Part 1* [online]. Medium. Available at: <https://medium.com/@saadkhan.cdz/introduction-to-unit-testing-in-android-getting-started-with-the-basics-part-1-4839c7a7a7f5> [Accessed 16 Apr. 2025].
- Revenue.ie, n.d. *Personal tax credits, reliefs and exemptions* [online]. Available at: <https://www.revenue.ie/en/personal-tax-credits-reliefs-and-exemptions/index.aspx> [Accessed 16 Apr. 2025].
- Revenue.ie, n.d. *PAYE Income Tax Return - End of Year Process* [online]. Available at: <https://www.revenue.ie/en/jobs-and-pensions/end-of-year-process/payee-income-tax-return.aspx> [Accessed 16 Apr. 2025].
- Revenue.ie, n.d. *Tax Relief Charts* [online]. Available at: <https://www.revenue.ie/en/personal-tax-credits-reliefs-and-exemptions/tax-relief-charts/index.aspx> [Accessed 16 Apr. 2025].
- Revenue.ie, n.d. *Tax Credits – Differences between Tax Credits, Reliefs, and Exemptions* [online]. Available at: <https://www.revenue.ie/en/personal-tax-credits-reliefs-and-exemptions/differences-between-tax-credits-reliefs-and-exemptions/tax-credits.aspx> [Accessed 16 Apr. 2025].
- Revenue.ie, n.d. *Tax Rate Bands* [online]. Available at: <https://www.revenue.ie/en/jobs-and-pensions/calculating-your-income-tax/tax-rate-band.aspx> [Accessed 16 Apr. 2025].



Atlassian, n.d. *Minimum Viable Product (MVP)* [online]. Available at: <https://www.atlassian.com/agile/product-management/minimum-viable-product#:~:text=The%20concept%20of%20the%20minimum,customers%20with%20the%20least%20effort.%E2%80%9D> [Accessed 18 Apr. 2025].

## 10. Appendix

### 10.1 Appendix A - User Stories

Title: Create an Account

Priority: High

User Story:

As a user, I want to create an account so that my data is saved securely.

Acceptance Criteria:

Given that a user is on the sign-up screen, when they enter valid details and submit, then their account is created, and they receive a confirmation.

Title: Log Multiple Hourly Jobs

Priority: High

User Story:

As a user, I want to log multiple hourly jobs to track my income accurately.

Acceptance Criteria:

Given that a user is on the job logging screen, when they add details of multiple hourly jobs, then the app saves and displays their logged jobs.

Title: Add Expenses

Priority: High

User Story:

As a user, I want to add my expenses so that I can manage deductions effectively.

Acceptance Criteria:

Given that a user is on the expense entry screen, when they input expense details and save, then the app records the expenses for future reference.

Title: View Estimated Tax Owed

Priority: High

User Story:

As a user, I want to view my estimated tax owed so that I can understand my financial obligations.

Acceptance Criteria:

Given that the user has logged income, when they navigate to the tax summary screen, then the app displays estimated taxes owed.

Title: Check Refund Eligibility

Priority: Medium

User Story:

As a user, I want to check if I'm eligible for refunds and reclaim any overpaid taxes.

Acceptance Criteria:

Given that the user has logged income and expenses, when they check refund eligibility, then the app calculates and informs them if they qualify for a refund.

Title: Access Income Data Offline

Priority: Medium

User Story:

As a user, I want to access my income data offline so that I can view my details anytime.

Acceptance Criteria:

Given that the user has saved income data, when they access the app offline, then previously saved income data is available.

Title: Track Weekly Hours

Priority: Medium

User Story:

As a user, I want to track my weekly hours to estimate my monthly tax liabilities.

Acceptance Criteria:

Given that the user logs work hours when they check their weekly summary, then the app provides an estimate of monthly tax liabilities.

Title: View Monthly Income Overview

Priority: Medium

User Story:

As a user, I want to view an overview of my monthly income so that I can budget better.

Acceptance Criteria:

Given that the user has logged income, when they open the income summary page, then they see a breakdown of their monthly earnings.

Title: Receive Refund Notifications

Priority: Medium

User Story:

As a user, I want to receive notifications for potential refunds to avoid missing them.

Acceptance Criteria:

Given that the user has refund eligibility, when a refund opportunity is detected, then they receive a notification.

Title: Calculate Provisional Taxes

Priority: Medium

User Story:

As a user, I want to calculate provisional taxes at year-end to plan for tax obligations.

Acceptance Criteria:

Given that the user has logged income and expenses, when they check provisional tax calculations, then the app provides an estimated tax amount.

Title: Track Tuition-Related Expenses

Priority: Medium

User Story:

As a student, I want to track tuition-related expenses to claim deductions.

Acceptance Criteria:

Given that the user is on the expense tracking page, when they add a new tuition-related expense, then the app records and categorises it under tuition for deduction purposes.

Title: View Tax Breakdown Per Job

Priority: Medium

User Story:

As a part-time worker, I want to view my tax breakdown per job for detailed insights.

Acceptance Criteria:

Given that the user has logged multiple jobs, when they access the tax breakdown screen, then the app displays a detailed breakdown of taxes for each job separately.

Title: View Previous Years' Tax Summaries

Priority: Medium

User Story:

As a user, I want the option to view previous years' tax summaries to analyse trends.

Acceptance Criteria:

Given that the user has past tax summaries saved, when they select a previous year, then the app displays the corresponding tax summary.

Title: Access Tax Resources and Tips

Priority: Low

User Story:

As a student, I want to access tax resources and tips within the app to learn more about managing finances.

Acceptance Criteria:

Given that the user navigates to the "Tax Resources" section, when they browse available topics, then the app displays relevant articles and guides.

Title: Customise Notifications

Priority: Low

User Story:

As a user, I want to customise my notifications to receive only relevant financial updates.

Acceptance Criteria:

Given that the user accesses notification settings, when they toggle specific categories on or off, then they only receive notifications for selected categories.

Title: Download Tax Summaries

Priority: Medium

User Story:

As a part-time worker, I want to download my tax summaries for my records.

Acceptance Criteria:

Given that the user is on the tax summary page, when they select “Download”, then the app generates and downloads the summary as a PDF file.

Title: Secure Data Storage

Priority: High

User Story:

As a user, I want my data to be securely stored, ensuring that my information is safe.

Acceptance Criteria:

Given that the user has stored financial data, when they use the app, then their data is encrypted and protected against unauthorised access.

Title: User-Friendly Interface for Tax Calculations

Priority: High

User Story:

As a user, I want a user-friendly interface that simplifies tax calculations.

Acceptance Criteria:

Given that the user is calculating taxes, when they enter relevant income and deductions, then the app provides a simple step-by-step calculation experience.

Title: Access Customer Support

Priority: High

User Story:

As a user, I want access to customer support to resolve any app-related issues.

Acceptance Criteria:

Given that the user has an issue, when they navigate to the support section, then they see options for live chat, email support, and FAQs.

Title: Delete Account

Priority: High

User Story:

As a user, I want to be able to delete my account in case I'm not interested in using the app anymore.

Acceptance Criteria:

Given that the user navigates to account settings, when they select “Delete Account” and confirm, then the app removes their account and all associated data permanently.

## 10.2 Appendix B - Survey Questions

For a better understanding of the target audience and their challenges with income tracking and tax filing, a user survey was done. The questions aim to assess users’ employment type, how many part-time jobs they have, tax knowledge, refund habits, and current income tracking methods. They were also asked whether they use any tools (apps or websites) to manage their taxes and if they are aware of tax credits and reliefs available to part-time workers.

The survey helped validate assumptions that many users struggle with financial management and would benefit from a mobile tool like SmartWage. The majority expressed interest in a tax calculator app that tracks income, estimates refunds, and provides clarity on tax credits and relief eligibility.

What is your current employment status? (Select all that apply)

43 responses

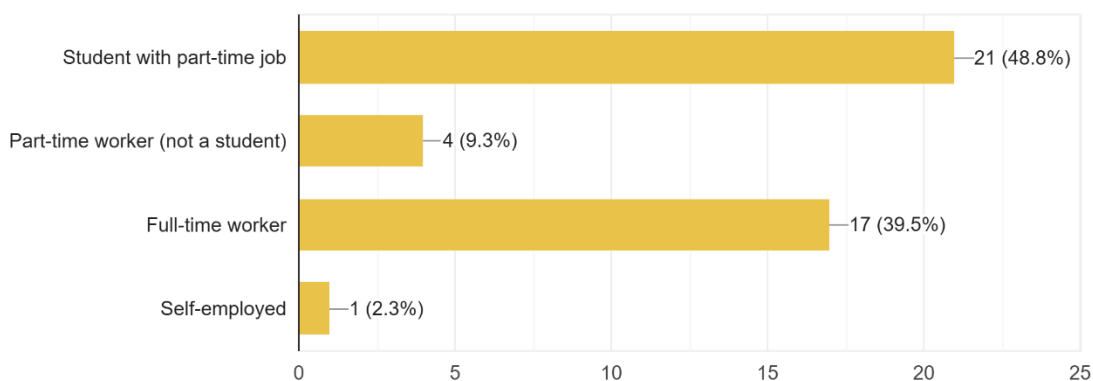


Image B.1 - Screenshot of question - “What is your current employment status?”

How many part-time jobs do you currently have?

43 responses

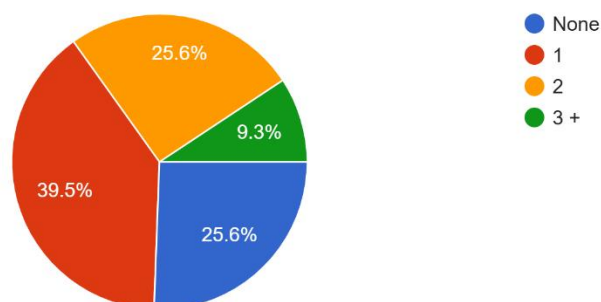


Image B.2 - Screenshot of question - "How many part-time jobs do you currently have?"

How many hours do you typically work in a week? (total across all jobs)

43 responses

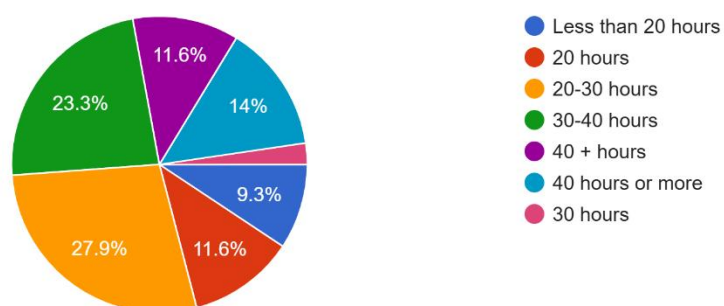


Image B.3 - Screenshot of question - "How many hours do you typically work in a week?"

What is your average hourly wage?

43 responses

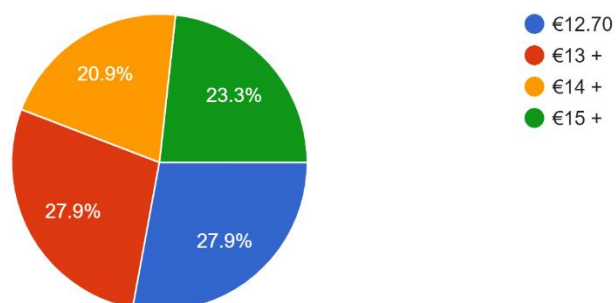


Image B.4 - Screenshot of question - "What is your average hourly wage?"

Do you have any additional sources of income? (Select all that apply)

43 responses

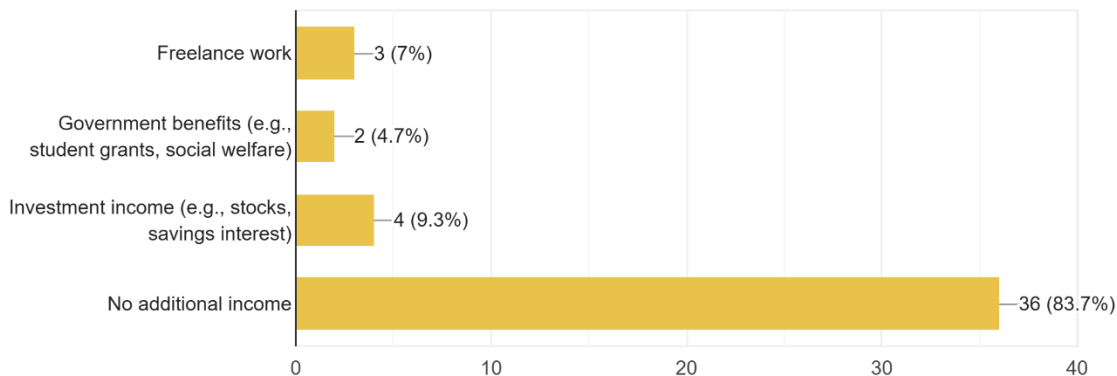


Image B.5 - Screenshot of question - "Do you have any additional source of income?"

How do you usually track your income?

43 responses

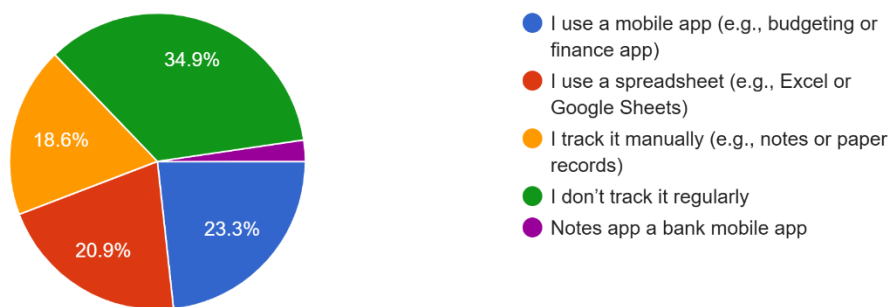


Image B.6 - Screenshot of question - "How do you usually track your income?"

Do you claim your tax refund every year?

43 responses

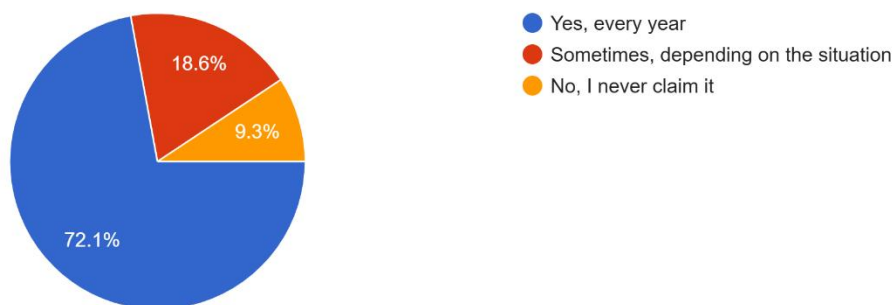


Image B.7 - Screenshot of question - "Do you claim your tax refund every year?"



If you don't claim your tax refund every year, why not? (Select all that apply)

43 responses

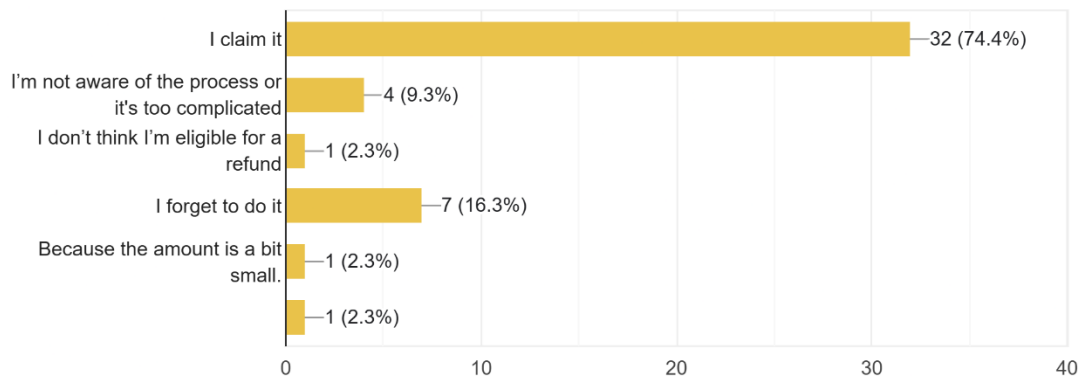


Image B.8 - Screenshot of question - "If you don't claim you tax refund every year, why not?"

How do you usually file your taxes and claim refunds?

43 responses

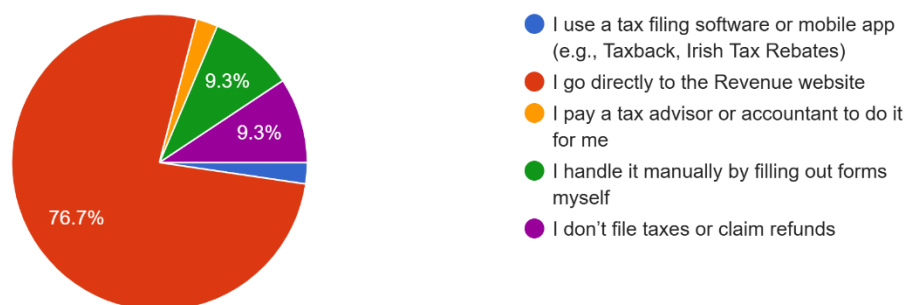


Image B.9 - Screenshot of question - "How do you usually file your taxes and claim refunds?"

If you use a web or a mobile app for tax filing, which one do you use? (Select all that apply)

43 responses

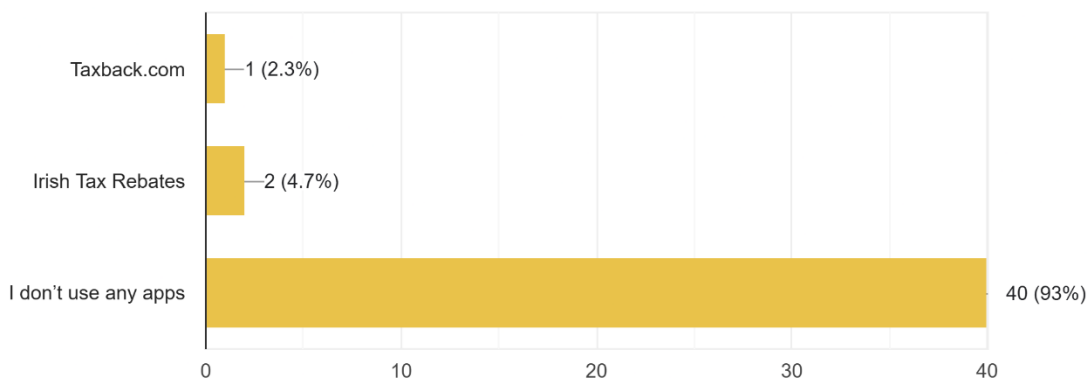


Image B.10 - Screenshot of question - "If you use web or app for tax filling, which one do you use?"

Are you aware of any tax reliefs or credits that you can claim as a part-time worker or student?  
43 responses



Image B.11 - Screenshot of question - “Are you aware of any tax reliefs or credits that you can claim as a part-time worker or student?”

Which tax reliefs or credits have you claimed in the past? (Select all that apply)  
43 responses

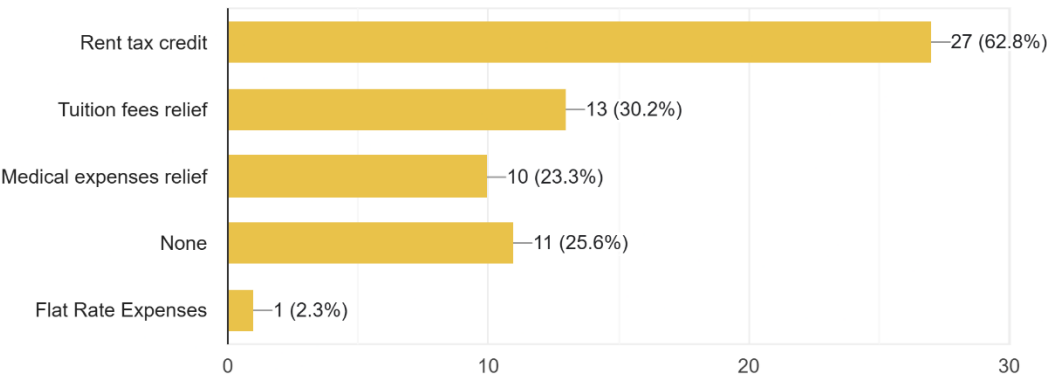


Image B.12 - Screenshot of question - “Which tax reliefs or credits have you claimed in the past?”

On a scale of 1 to 5, how easy do you find the process of filing taxes and claiming refunds?  
43 responses

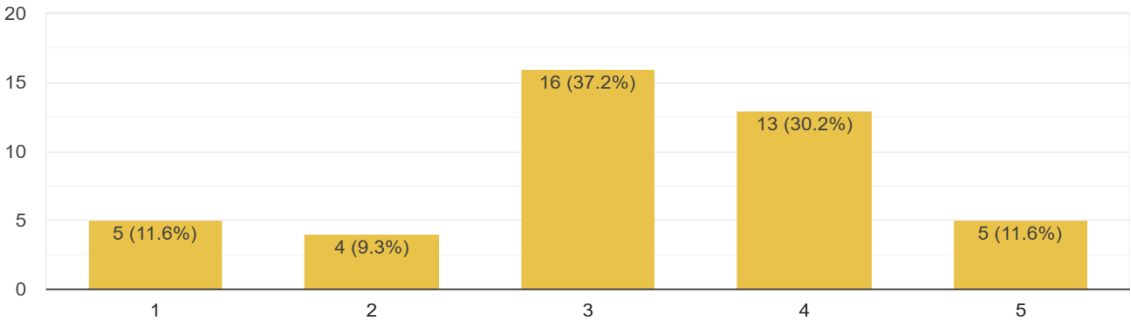


Image B.13 - Screenshot of question - “On scale of 1 to 5, how easy do you find the process of filling taxes and claiming refunds?”

Would you be interested in using an informative mobile app that helps track your income, calculate your taxes, and estimate your refunds automatically?

43 responses

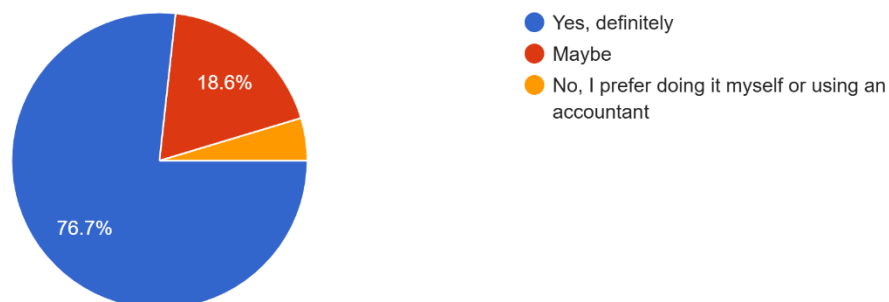


Image B.14 - Screenshot of question - "Would you be interested in using an informative mobile app that helps track your income, calculate your taxes, and estimate your refunds automatically?"

What features would you find most useful in a tax-tracking app? (Select all that apply)

43 responses

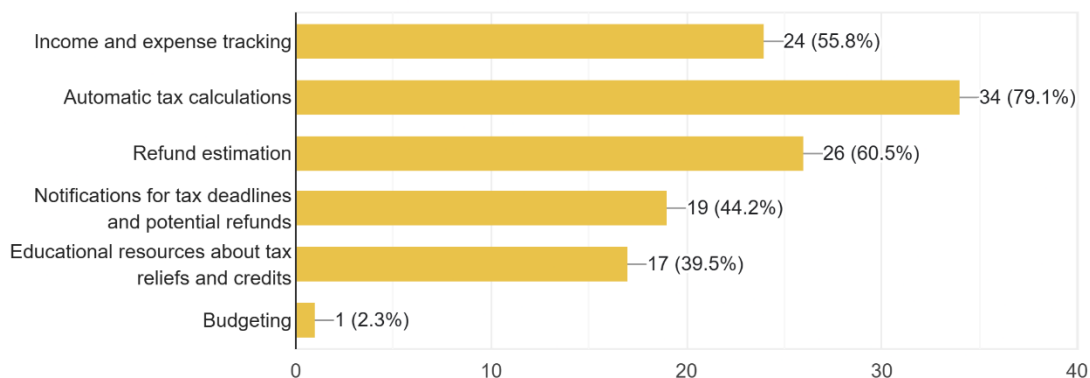


Image B.15 - Screenshot of question - "What feature would you find most useful in a tax-tracking app?"

## 10.3 Appendix C - Survey Suggestion

They were invited to share suggestions for future app features.

Notable ideas included:

- Integration with payslips
- Automated reminders for tax submissions
- Simpler language in tax-related screens
- Support for multiple income streams

These insights were valuable during development and have been incorporated into our Future Improvements roadmap in Chapter 6.

Do you have any suggestions for features or improvements you'd like to see in a tax-tracking app?  
(optional)

10 responses

- They should notify about those taxes refunds
- A Reminder feature
- None.
- Bar code readers
- An intuitive app would be useful and appreciated
- I'd like to be informed of the things I can be given tax credits or even be refunded. I'd also like to know about deadlines for submissions.
- A way that we can see the amounts of taxes paid and incomes received in the year or weekly and projection if there's any amount to claim it back
- Reminders
- Advise about how to reduce the taxes.
- It would be great if you could calculate an estimate of how much you would have to receive.

Image C.1 - Screenshot of question - “Do you have any suggestion for features or improvements you would like to see in tax-tracking app?”

## 10.4 Appendix D - Survey Frustration

Participants were also asked to describe their biggest frustration with the current tax filing process. Common answers were:

- Lack of clarity on how PAYE, PRSI, and USC are calculated
- Struggle to understand eligibility for tax refunds
- The need to rely on external professionals or services to file taxes

These frustrations underline the core problem SmartWage seeks to address, making personal tax management more accessible and understandable for part-time workers and students.

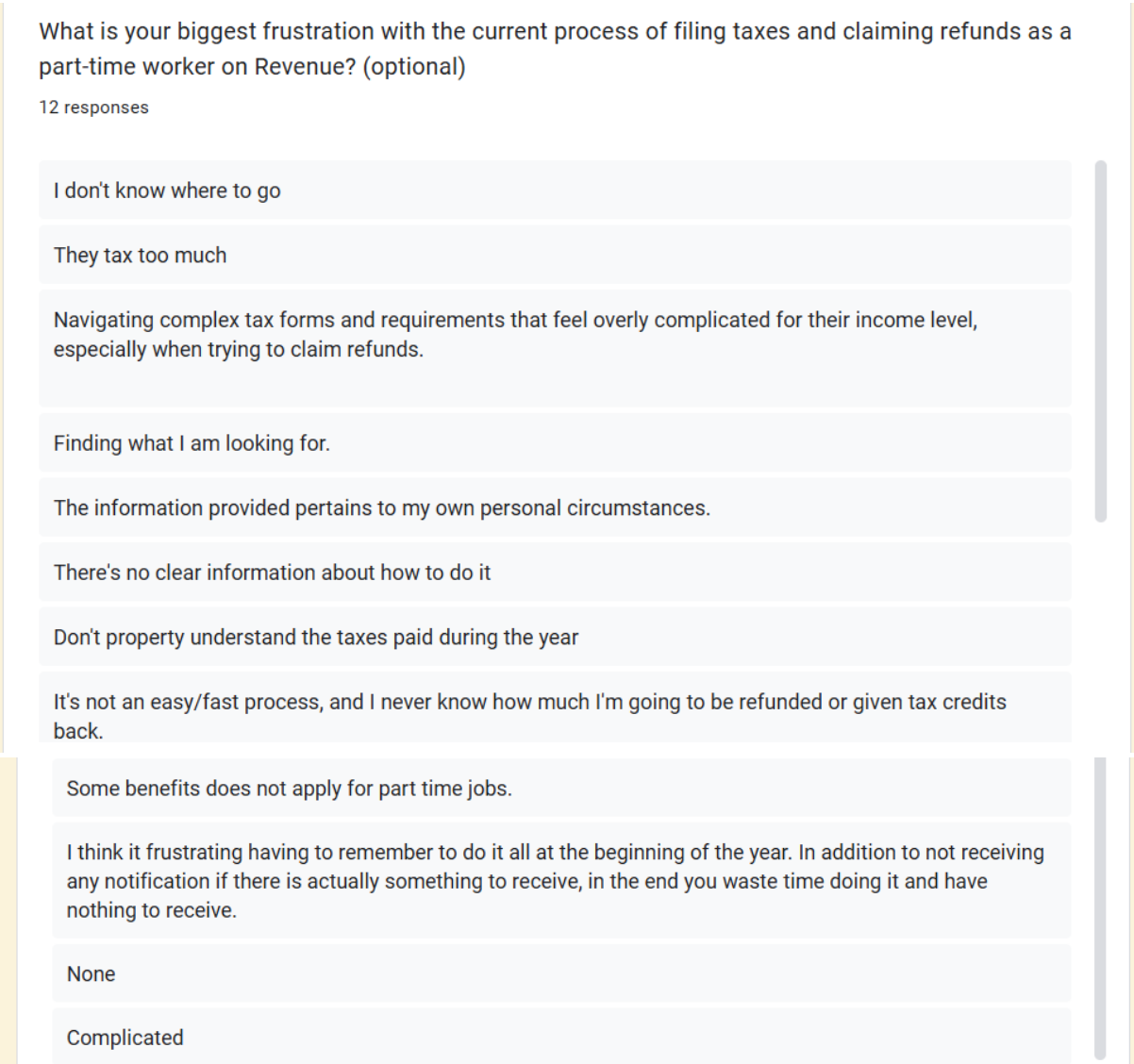


Image D.1 - Screenshot of question - "What is your biggest frustration with the current process of filling taxes and claiming refunds as a part-time worker or Revenue?"

## 10.4 Appendix E - Backlog GitHub Projects

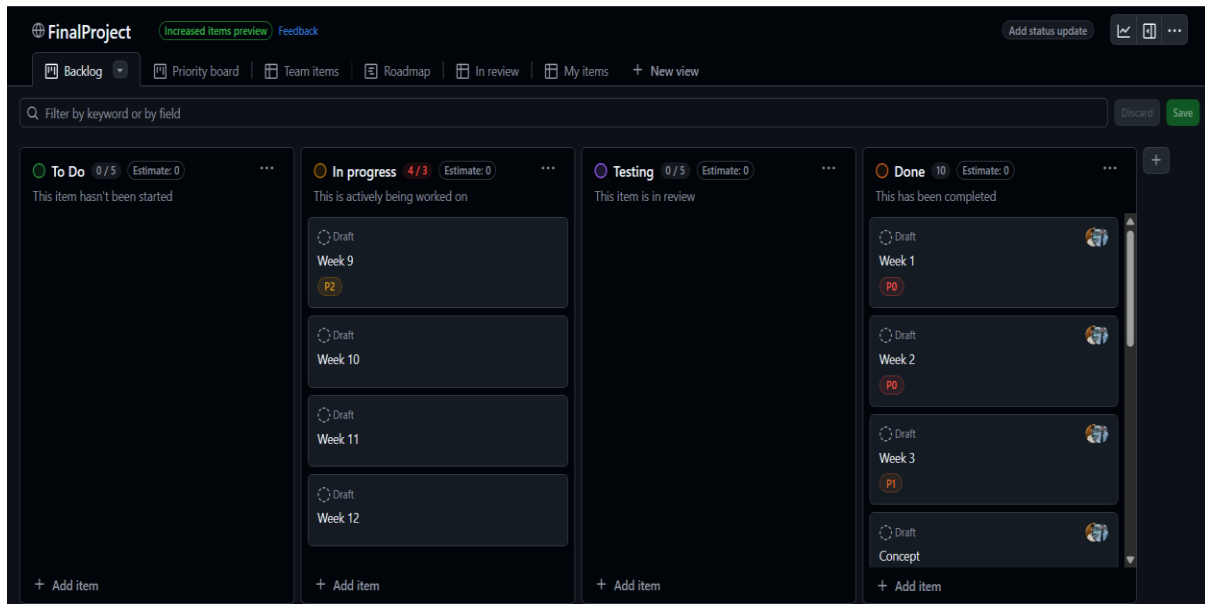


Image E.1 - Screenshot of Smart Wage backlog board

### 10.4.1 Appendix E - Backlog Summary

The screenshot above shows the project backlog board, which was created and maintained throughout the project using GitHub Projects. It is like Trello, but once you only use one platform, it was an easy way to keep the project on time for tasks. The Kanban style visual was managed weekly, there the team usually track sprint progress and prioritise the activities using Agile methodology.

The board was categorised into four columns:

- To Do - Tasks that were not initialised yet.
- In Progress - Tasks that the team are working on.
- Testing - Features that the team left under review.
- Done - Completed tasks.

Each card task was labelled by week (e.g., week1, week2), including priority tags (P0= high, P1=medium, P2=low). That way the team could have better focus on incremental goals, control on better time estimation, also an iterative delivery mode.

This methodology supported collaboration and transparency on the process, as the team and the supervisor could monitor the progress. The structure was so important for the project delivery and all the changes made through the development during the semester.