

AGA

Advanced Guidance Assistance

Interim Report

TU858/DT282

BSc in Computer Science

(International)

**Lovely Joy Velasco Fernandez**

**C20305696**

**Bojan Bozic**

School of Computer Science

Technological University, Dublin

**04th November 2024**

Abstract

The Advanced Guidance Assistance (AGA) bot is a tool designed specifically to support individuals with ADHD in managing their daily tasks and routines. It acts as a personal assistant that provides structured guidance, reminders, and encouragement to help users stay focused and on track throughout their day. With features like customisable task lists, real-time alerts, and adaptive goal setting, AGA helps reduce stress, anxiety and improve time management, making it easier to build healthier habits and maintain productivity.

Its user-friendly interface and supportive approach aim to empower users to navigate their day more effectively and confidently.

Declaration

I hereby declare that the work described in this dissertation is, except where otherwise stated, entirely my own work and has not been submitted as an exercise for a degree at this or any other university.

Signed:

\_\_Lovely Fernandez\_\_\_\_

Lovely Joy Velasco Fernandez

04th November 2024

Acknowledgements

Body text

Table of Contents

[1. Introduction 7](#_Toc181627952)

[1.1. Project Background 7](#_Toc181627953)

[1.2. Project Description 7](#_Toc181627954)

[1.3. Project Aims and Objectives 7](#_Toc181627955)

[1.4. Project Scope 7](#_Toc181627956)

[1.5. Thesis Roadmap 7](#_Toc181627957)

[2. Literature Review 8](#_Toc181627958)

[2.1. Introduction 8](#_Toc181627959)

[2.2. Alternative Existing Solutions to Your Problem 8](#_Toc181627960)

[2.3. Technologies you’ve researched 8](#_Toc181627961)

[2.4. Other Research you’ve done 8](#_Toc181627962)

[2.5. Existing Final Year Projects 8](#_Toc181627963)

[2.6. Conclusions 8](#_Toc181627964)

[3. System Design 9](#_Toc181627965)

[3.1. Introduction 9](#_Toc181627966)

[3.2. Software Methodology 9](#_Toc181627967)

[3.3. Overview of System 9](#_Toc181627968)

[3.X. Other Sections 9](#_Toc181627969)

[3.X. Conclusions 9](#_Toc181627970)

[4. Testing and Evaluation 10](#_Toc181627971)

[4.1. Introduction 10](#_Toc181627972)

[4.2. Plan for Testing 10](#_Toc181627973)

[4.3. Plan for Evaluation 10](#_Toc181627974)

[4.4. Conclusions 10](#_Toc181627975)

[5. Prototype Development 11](#_Toc181627976)

[5.1. Introduction 11](#_Toc181627977)

[5.2. Prototype Development 11](#_Toc181627978)

[5.3. Other Sections 11](#_Toc181627979)

[5.4. Conclusions 11](#_Toc181627980)

[6. Project Management 12](#_Toc181627981)

[6.1. Project Plan 12](#_Toc181627982)

[6.1.1 Milestones & Roadmap 12](#_Toc181627983)

[6.1.2 Development Methodology 13](#_Toc181627984)

[6.1.3 Task Distribution 13](#_Toc181627985)

[6.2. Supervisor Engagement 13](#_Toc181627986)

[6.2.1 Meeting Availability 14](#_Toc181627987)

[6.3. Risk Management 14](#_Toc181627988)

[6.3.1. Technical Risks 14](#_Toc181627989)

[6.3.2. User Risks 14](#_Toc181627990)

[6.3.3. Project Management 14](#_Toc181627991)

[6.3.4. Legal and Regulatory Risks 15](#_Toc181627992)

[6.3.5. Steps for Effective Risk Management 15](#_Toc181627993)

[7. Project Plan for Next Phase 15](#_Toc181627994)

[7.1. Introduction 15](#_Toc181627995)

[7.3. Plans and Future Work 15](#_Toc181627996)

[7.3.1. GANTT Chart 15](#_Toc181627997)

[Bibliography 16](#_Toc181627998)

# 1. Introduction

AGA is an advanced guidance assistance developed for individuals with ADHD but could also be openly used by individuals struggling to maintain a balanced lifestyle, for example, students.

The main challenges highlighted

## Project Background

Some background and literature, start with an interesting fact or a newspaper item

## Project Description

An overview of the project

Include a diagram

## Project Aims and Objectives

Overall aim and some milestones along the way to achieve the aim

* 5-9 objectives

## Project Scope

Project scope, what the project isn’t about

## Thesis Roadmap

One sentence explaining what each of the following chapters is about.

# Literature Review

## 2.1. Introduction

In this chapter …

## 2.2. Alternative Existing Solutions to Your Problem

Software you’ve looked into

## 2.3. Technologies you’ve researched

Programming languages, operating systems, etc.

## 2.4. Other Research you’ve done

Domain specific research

## 2.5. Existing Final Year Projects

## 2.6. Conclusions

# 3. System Design

## 3.1. Introduction

## 3.2. Software Methodology

## 3.3. Overview of System

## 3.X. Other Sections

## 3.X. Conclusions

# 4. Testing and Evaluation

## 4.1. Introduction

## 4.2. Plan for Testing

## 4.3. Plan for Evaluation

## 4.4. Conclusions

# 5. Prototype Development

**As least 2 pages, but as many as you like (but lots of code samples).**

## 5.1. Introduction

## 5.2. Prototype Development

## 5.3. Other Sections

## 5.4. Conclusions

# 6. Project Management

This section of the report provides an overview of the project management approach utilised for this project. It covers the *project plan - milestone roadmap, development methodology and task distribution methods - supervisor and team engagement strategies,* as well as *risk management practices*.

Each component is discussed to highlight how structured planning, communication, and proactive risk mitigation were applied to achieve project objectives effectively.

## 6.1. Project Plan

### 6.1.1 Milestones & Roadmap

The project plan is divided into four quarters, with each semester consisting of two quarters. The first semester focuses on completing the development of the product, while the second semester is dedicated to the refinement of the product.

A screenshot of a computer

Description automatically generated

*Figure 6.1.1.A Milestones First Semester*



*Figure 6.1.1.B Milestones Second Semester*

The project roadmap is automatically generated in GitHub when creating iterations. By setting up these iterations, GitHub organises the tasks, deadlines, and progress into a clear, visual roadmap. This automated roadmap provides an overview of the project's timeline, making it easy to track each phase’s progress and ensuring that the project stays aligned with its planned schedule.

*Find the roadmap on AGA’s Project at https://github.com/users/vedez/projects/2.*

### 6.1.2 Development Methodology

An Agile approach will be used throughout the project, which means work will be organised into iterative cycles, allowing for continuous improvement and adaptability. Agile promotes flexibility by focusing on incremental progress through manageable "sprints," or short cycles.

*“Agile teams also believe in continuously evaluating project plans and objectives. As a result, they are highly flexible and can quickly adjust to change…Adopting the Agile method will allow you to operate in a detail-oriented manner.”* (Team, 2022)

Each sprint includes planning, task allocation, execution, review, and adjustment phases. This iterative method allows a quick response to feedback, address issues as they arise, and enhance project quality progressively.

### 6.1.3 Task Distribution

Tasks are organised by priority and distributed across the following stages to maintain a streamlined workflow.

*Find the backlog of tasks on AGA’s Project at https://github.com/users/vedez/projects/2/views/1.*

* **To Do**

This stage includes all pending tasks, organised by labels such as *Stories, Tasks, and Bugs*. Each item represents a possible issue or feature for the project.

Uncompleted tasks stay in the backlog until allocated catch-up time.

* **In Progress**

Tasks are tackled one at a time to help maintain focus and prevent overwhelming scenarios. Stories are broken down into manageable tasks, which are completed individually to maintain a clear path from "to do" to "done."

* **Review & Testing**

Completed features are submitted in a different branch, where they undergo review and testing. Once a feature meets standards, it is merged into the main project branch, reducing bugs and maintaining quality.

* **Done**

Tasks are marked as "done" once they meet all requirements and quality standards. In this project, the definition of "done" means that the feature fulfils all specified requirements and has little to no bugs. Minor bugs that do not significantly impact the project can be logged as separate bug tasks to be addressed during review or allocated catch-up time.

## 6.2. Supervisor Engagement

Dr. Bojan Božić is an Assistant Lecturer in Computer Science at TU Dublin with extensive expertise in Semantic Web, Machine Learning, and Data Science. He will be supervising the development of my fourth-year project, *AGA*.

*More information could be found at* [*https://www.tudublin.ie/explore/faculties-and-schools/computing-digital-data/school-of-computer-science/people/academic-staff/bojanbozic.php*](https://www.tudublin.ie/explore/faculties-and-schools/computing-digital-data/school-of-computer-science/people/academic-staff/bojanbozic.php)*.*

### 6.2.1 Meeting Availability

|  |  |
| --- | --- |
| Tuesday | 14:00 – 15:00 |
| Wednesday | 11:00 – 15:00 |
| Thursday | 11:00 – 15:00 |

#### 6.2.1.1 Types of Meeting

* **Catch Up**
  + A quick catch-up meeting regarding what has been completed during the week. This could either be delivered via on-site/online meeting or re-cap document sent via email.
  + Time duration: 15 minutes max.
* **Review**
  + This meeting reviews bigger accomplishments - *features, testing, etc… -* to discuss about what has been completed, possible changes to apply, and general feedback of these tasks.
  + Time duration: 30 minutes max.
* **Retrospective**
  + This meeting opens feedback regarding communication and the handling of the project if any aspects need to be changed or continued. This is important to ensure morale and strengthens communication of the team.

## 6.3. Risk Management

### 6.3.1. Technical Risks

* **Software Bugs and Errors**

Mitigated through regular testing and continuous review of the issue board. Non-functioning features should be flagged immediately and either fixed within the current iteration or added to the backlog for scheduled review and correction.

* **Version Control**

Use branches to manage different versions of the code, merging only after code has passed review to reduce bugs in the main project. Labelling “done” in the definition stated in *6.1.3 Task Distribution.*

* **Security and Authorisation**

Implement Object-Oriented Programming (OOP) principles and assign appropriate authorisation levels to ensure secure, manageable code.

### 6.3.2. User Risks

* **User Satisfaction and UX Failures**

Addressed through dedicated UX testing cases that prioritise user feedback to enhance usability. This is the top priority during the second semester for code implementation.

* **User Communication**

Engaging with users to understand their needs is essential, focusing on features that benefit their experience and usability of the application.

### 6.3.3. Project Management

* **Time Constraints**

Each task or story is estimated for time, with allowances for unexpected delays by incorporating buffer time into the schedule.

* **Scope Creep**

Clearly define the project scope, prioritising core features. Additional requests or features are added to the backlog to address only if time allows, ensuring focus and avoiding overloading tasks.

* **Freeze Point**

Establish a *"freeze point"* where no new features or changes can be added, solidifying the final scope for efficient completion.

### 6.3.4. Legal and Regulatory Risks

* **Data Privacy Compliance**

Ensure all user data handling complies with GDPR and other relevant data privacy laws.

* **Age Restrictions and Safety**

Consider age restrictions, ensuring that the product is safe and accessible for younger audiences. Additional laws regarding online safety and accessibility may also be reviewed.

### 6.3.5. Steps for Effective Risk Management

1. **Risk Identification**: List potential risks across technical, user, project management, and regulatory areas.
2. **Risk Assessment**: Evaluate the likelihood and impact of each risk.
3. **Risk Prioritisation**: Prioritise high-risk items that have both high likelihood and high impact.
4. **Mitigation Planning**: Develop strategies to manage risks, such as additional testing phases.

# 7. Project Plan for Next Phase

## 7.1. Introduction

## 7.3. Plans and Future Work

### 7.3.1. GANTT Chart

# Bibliography

Team, T. (2022, June 29). Stages in the Life-cycle of Agile Development Process. *TheCodeWork*. https://thecodework.com/blog/stages-in-the-life-cycle-of-agile-development-process/