HowTo

How to do anything in the world.

Problem

People often have to wing their way through life, not knowing how to do things. This means that either, they don't do things, or they do them badly which can lead to a lot of frustration and wasted time and resources.

Solution

I propose a web application that will allow those who have done something before to write a structured guide on how to do the said thing, with pictures and videos where necessary. Those who need help with regards to doing the specific thing can then come to the web application and search for the guide on how to do the thing.

In a situation where the said thing is not available, the user can request for a guide on how to do the thing. The request will be sent to the community of users who have done the thing before, and they can then write a guide on how to do the thing.

The web application will also have a rating system where users can rate the guides on how helpful they were. This will help other users to know which guides are the best to use.

And lastly, users will be able to pay the guide writers for their work if they so wish. This will encourage more people to write guides on how to do things.

Some users may not want to follow the guide themselves, so they can hire someone to do the thing for them. This is where the web application will come in. The web application will have a marketplace where users can hire people to do things for them. The people who are hired will then write a guide on how they did the thing, and this guide will be available to other users who want to do the thing. Obviously, the person who was hired will be paid for their work. Users will then review the person who was hired, and this will help other users to know who to hire.

Functional Requirements

1. Users should be able to create an account on the web application.
2. Users should be able to write a guide on how to do something.
    2.1. The guide should be structured in a way that is easy to follow.
    2.2. The guide should have pictures and videos where necessary.
    2.3. The guide should be tagged with the thing that it is about.
    2.4. The guide should have steps that represent an action that the user should take at each point in the guide.
3. Users should be able to search for a guide on how to do something.
4. Users should be able to request a guide on how to do something.
5. Users should be able to rate a guide on how helpful it was.

6. Users should be able to pay the guide writers for their work.
7. Users should be able to hire someone to do something for them.
8. Users should be able to write a guide on how they did something for someone else.
9. Users should be able to review the person who was hired to do something for them.
10. The web application should have a marketplace where users can hire people to do things for them.
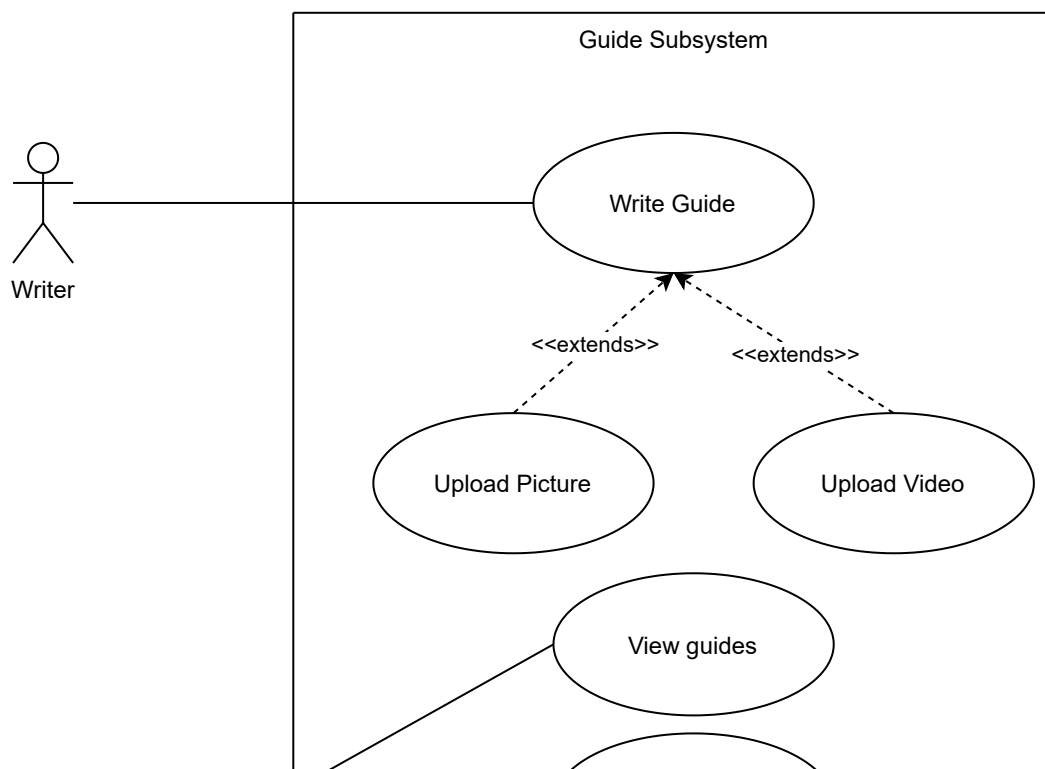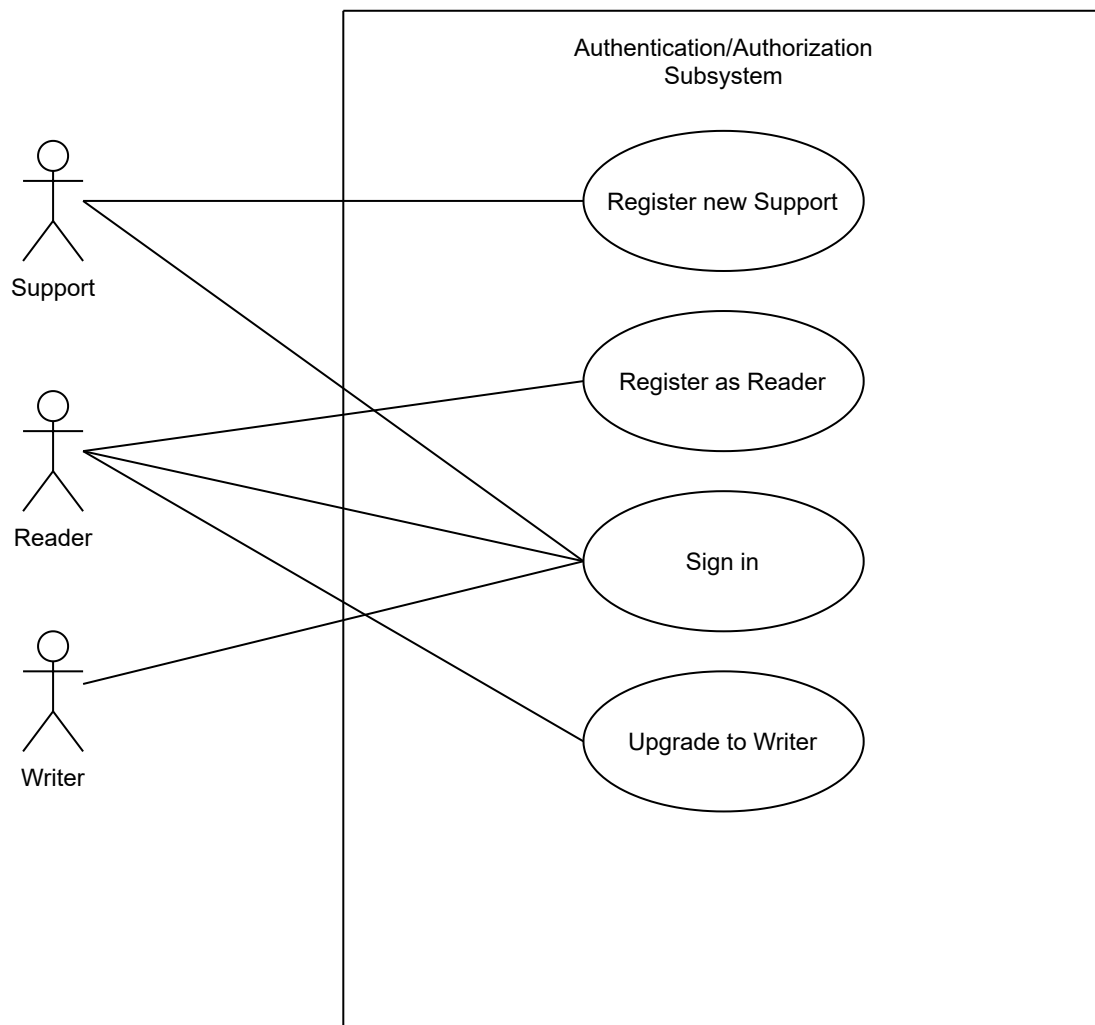
Non-Functional Requirements

1. The web application should be easy to use, and this will be measured by the number of users who are able to write a guide on how to do something.
2. The web application should be fast, and this will be measured by the time it takes for a user to search for a guide on how to do something.
3. The web application should be secure, and this will be measured by the number of security breaches that occur.
4. The web application should be reliable, and this will be measured by the number of times the web application goes down.
5. The web application should be scalable, and this will be measured by the number of users who are able to use the web application at the same time.
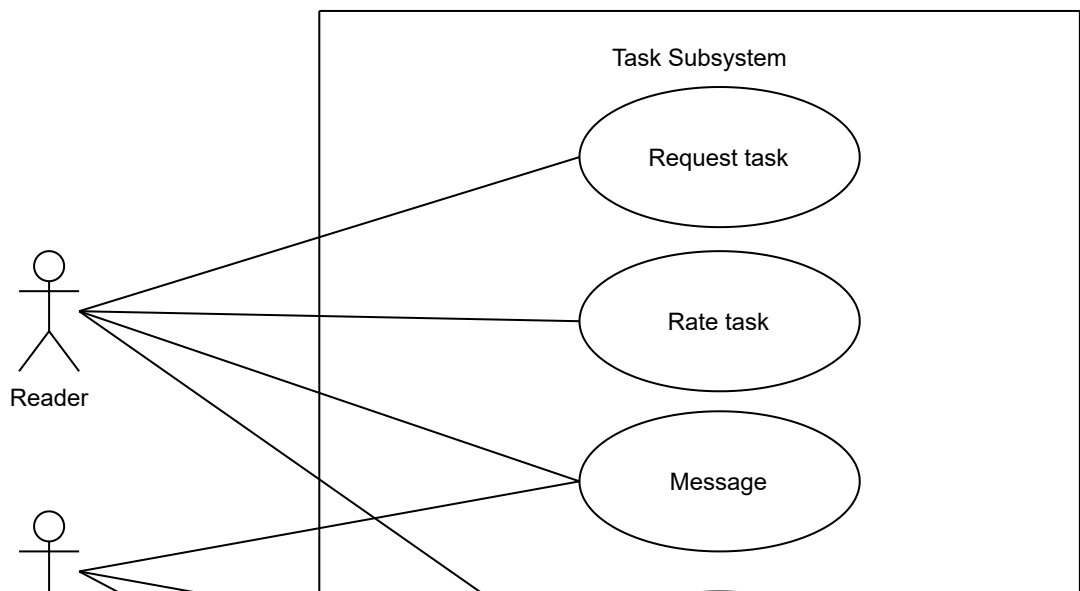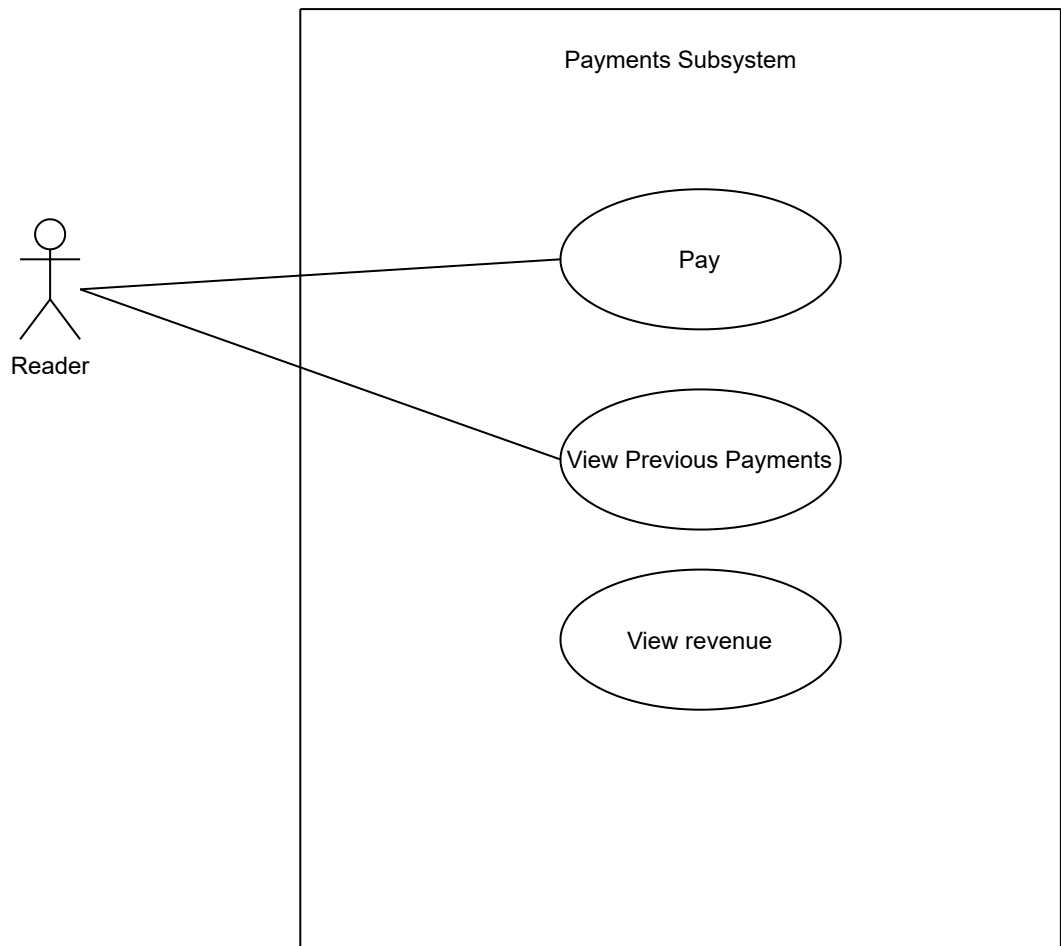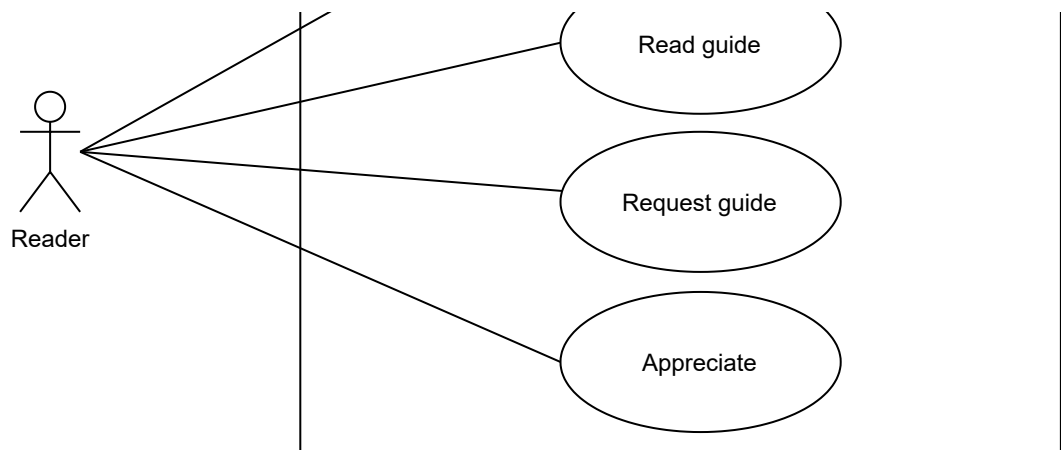
Users

1. Guide Writers: Users who have done something before and want to write a guide on how to do the thing.
2. Guide Readers: Users who need help with doing something and want to read a guide on how to do the thing.
3. Guide Requesters: Users who need help with doing something and want to request a guide on how to do the thing.
4. Guide Raters: Users who want to rate a guide on how helpful it was.
5. Guide Payers: Users who want to pay the guide writers for their work.
6. Thing Doers: Users who want to hire someone to do something for them.
7. Thing Doer Writers: Users who have been hired to do something and want to write a guide on how they did the thing.
8. Thing Doer Reviewers: Users who want to review the person who was hired to do something for them.
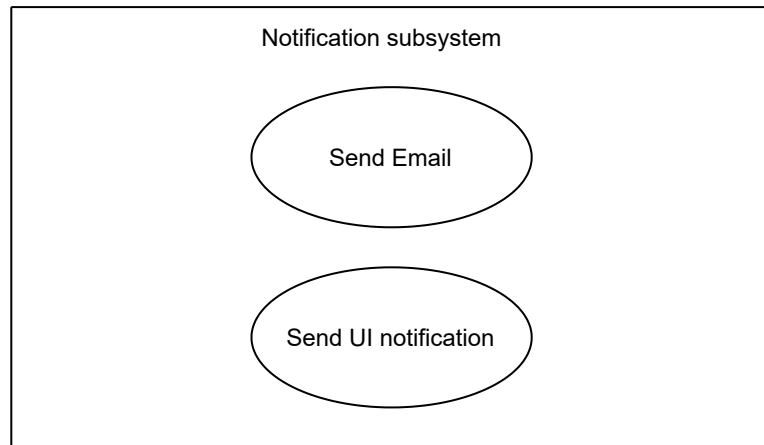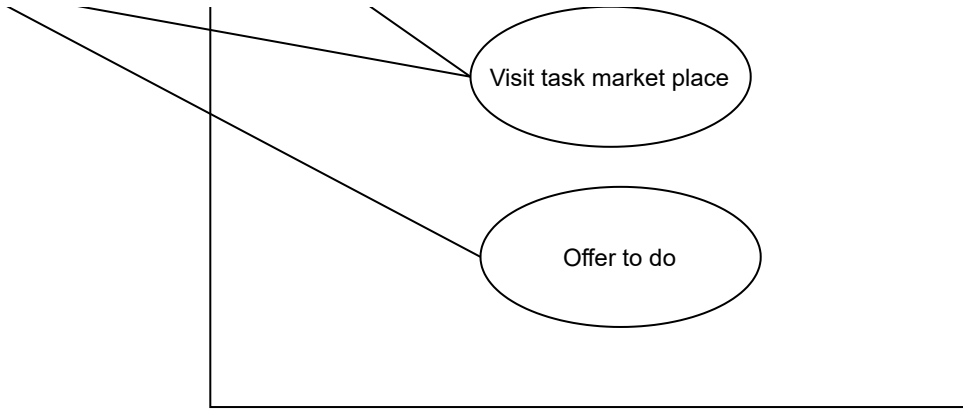
The web application should have a clear distinction between the different roles that a user can have. So that a user can easily switch between the different roles that they have. The main dinstinct user types are Writer[W], Writer Probation [WP], Reader[R], and the Support[S] users. The assumption that we will make is that the Writer can write the guide and execute tasks, but they can also act as a Reader, The Reader cannot write reviews, or execute tasks. Reader can only read the guides and execute their own tasks. The Writer user can do all that a Reader can do and also write reviews, execute tasks, and write guides. The Support user can do all that can be done in the system, but they are only people who work for the company that owns the web application. Readers can upgrdade to Writers based on the amount of guides that they have read, they can be put on probation by allowing them to write guides that will be reviewed by the Support users. The Support users can also downgrade the Writers to Readers if they are not writing good guides.

Use Cases

Authentication/Authorization Subsystem

- Support → Register new Support
- Support → Sign in
- Reader → Register as Reader
- Reader → Sign in
- Reader → Upgrade to Writer
- Writer → Sign in

Guide Subsystem

- Writer → Write Guide
- Upload Picture <<extends>> Write Guide
- Upload Video <<extends>> Write Guide
- View guides

Read guide

Request guide

Appreciate

Reader

Payments Subsystem

Pay

View Previous Payments

View revenue

Reader

Task Subsystem

Request task

Rate task

Message

Reader

Writer

Visit task market place

Offer to do

Notification subsystem

Send Email

Send UI notification

Domain Model

Task

PK	Id
	description
	owner
	guideId
	views
	amount
	acceptedAmount
	timeFrame
	status

Guide

PK	Id
	title
	description
	writer
	views
	dateTimeCreated

GuideSteps

PK	Id
	guideId
	value
	description

Payment

PK	Id
	payerId
	beneficiaryId
	amount
	dateTimePaid
	for

Reviews

PK	Id
	taskId
	reviewerId
	review
	dateTimeReviewed

upvotes

PK	Id
	guideId/reviewId
	userId
	type(guide/review)
	dateTimeVoted

Media

PK	Id
	filename
	dateTimePosted
	type
	guideStepId

Messages

PK	Id
	taskId
	senderId
	receiverId
	dateTimeSent

User Stories / User Flows as Activity Diagrams

Technology Stack

1. Frontend: Next.js
2. Backend: ABP Framework
3. Database: Microsoft SQL Server
4. Hosting: Azure App Service (Backend) + Vercel (Frontend)
5. Payment Gateway: Payfast
6. Video Hosting: Azure Blob Storage
7. Image Hosting: Azure Blob Storage
8. Search: Azure Cognitive Search ??
9. Email: Azure Communication Services

10. Authentication: Azure Active Directory ??
11. Version Control: GitHub