

# BANCO VVBA



**Realizado por:**

MIGUEL ANGEL ROMALDE DORADO

**Dirigido por:**

JUAN ANTONIO ORTEGA RAMIREZ

# Tabla de contenidos

1.	INTRODUCCIÓN	4
1.1	ABSTRACT	4
1.2	AGRADECIMIENTOS	5
2.	ESTUDIO DEL ARTE	5
2.1	Tecnologías de FrontEnd	5
2.1.1	React	5
2.1.2	Angular	6
2.1.3	Vue.js	6
2.1.4	Svelte	7
2.1.5	Ember.js	7
2.1.6	Backbone.js	7
2.1.7	Preact	8
2.2	Tecnologías del BackEnd	8
2.2.1	Node.js	8
2.2.2	Django	8
2.2.3	Ruby on Rails	9
2.2.4	Spring boot	9
2.2.5	Express.js	9
2.2.6	Flask	10
2.2.7	ASP.NET Core	10
2.3	Selección de tecnologías	10
2.3.1	FrontEnd	11

2.3.2	BackEnd	12
3.	ALCANCE DEL PROYECTO	13
3.1	Elicitación de requisitos del producto	13
3.1.1	Requisitos de información	13
3.1.2	Requisitos funcionales	14
3.1.3	Requisitos no funcionales	15
3.1.3.1	Requisitos de calidad	15
3.1.3.2	Requisitos de implementación	15
3.1.3.3	Requisitos de seguridad	16
3.2	Riesgos	17
3.3	Viabilidad	18
3.4	Objetivos de la aplicación	18
4.	VERSIÓN DE COSTES	19
4.1	Coste salarial del desarrollo	19
4.2	Costes derivados del desarrollo	19
4.3	Costes de licencias	20
4.4	Coste del material de trabajo	21
4.5	Desglose total de los costes de desarrollo	22
4.6	Reservas de contingencia	22

## Lista de tablas

Tabla 1. Requisitos de información	14
Tabla 2. Requisitos funcionales	15
Tabla 3. Requisitos de calidad	15
Tabla 4. Requisitos de implementación	15
Tabla 5. Tabla de riesgos	18
Tabla 6. Coste de desarrollo del equipo	19
Tabla 7. Coste total salarial del desarrollo del proyecto	19
Tabla 8. Costes derivados en el desarrollo	20
Tabla 9. Costes derivados en el desarrollo totales	20
Tabla 10. Coste de licencias mensual	20
Tabla 11. Costes de licencias totales	21
Tabla 12. Precio material de trabajo	21
Tabla 13. Amortización de los equipos	21
Tabla 14. Amortización durante el desarrollo	21
Tabla 15. Costes totales del desarrollo	22
Tabla 16. Reservas de contingencia	22

## 1. INTRODUCCIÓN

Este trabajo pretende analizar y explorar alternativas disponibles en el mercado tecnológico ayudando y facilitando a la población, a entender cómo funciona una aplicación de banco sin ningún riesgo desde la comodidad de su hogar.

Por otra parte, para completar el caso de estudio y objetivo del trabajo en su forma práctica, se busca desarrollar una aplicación para la plataforma de escritorio Windows, de forma colateral, con la aplicación se pretende fomentar la independencia de los usuarios a la hora de trabajar con una aplicación de banco y reducir el miedo a la utilización de aplicaciones del mismo estilo.

Cabe destacar que se usarán tecnologías actuales el cual ayuden al propietario de la aplicación a su entrada a diferentes empresas. Posteriormente se hará un estudio de dichas tecnologías y se hará una selección acorde a los requerimientos.

### 1.1 ABSTRACT

This work aims to analyse and explore alternatives available on the technology market, helping and facilitating the population to understand how a banking application works without any risk from the comfort of their home.

On the other hand, to complete the study case and objective of the work in its practical form, the aim is to develop an application for the Windows desktop platform, collaterally, with the application it is intended to promote the independence of users when working with a banking application and reduce the fear of using applications of the same style.

It should be noted that current technologies will be used which will help the owner of the application to enter different companies. Subsequently,

a study of these technologies will be carried out and a selection will be made according to the requirements.

## 1.2 AGRADECIMIENTOS

En primer lugar, me gustaría agradecer a mi tutor, Juan Antonio Ortega Ramírez, por haber tutorizado este proyecto y por la enorme paciencia y flexibilidad que ha tenido conmigo. Me ha sabido guiar y darme respuesta a las necesidades que surgían a lo largo del desarrollo de este.

También, me gustaría agradecer a todos aquellos que, voluntariamente, participaron en la encuesta inicial del proyecto y en algunas de las pruebas.

Por último, gracias a mi familia y amigos que han estado apoyándome en momentos donde creía que no iba a conseguir finalizarlo a tiempo y me daban ganas de rendirme, es gracias ellos que este proyecto ha salido adelante.

## 2. ESTUDIO DEL ARTE

En este apartado vamos a hacer un estudio del arte con las 7 tecnologías más usadas actualmente en las empresas.

### 2.1 Tecnologías de FrontEnd

#### 2.1.1 React

 **Descripción:** Biblioteca de JavaScript para construir interfaces de usuario.

 **Características:**

- Componentes reutilizables.
- Virtual DOM para mejorar el rendimiento.
- Amplio ecosistema de herramientas y bibliotecas.

- ✚ **Ventajas:** Flexibilidad, gran comunidad, buen rendimiento.
- ✚ **Desventajas:** Puede ser complejo de aprender para principiantes debido a su ecosistema.

### 2.1.2 Angular

- ✚ **Descripción:** Framework de aplicaciones web basado en TypeScript.
- ✚ **Características:**
  - Estructura MVC.
  - Inyección de dependencias.
  - Herramientas integradas como Angular CLI.
- ✚ **Ventajas:** Estructurado, soporte para aplicaciones grandes, dos vías de enlace de datos.
- ✚ **Desventajas:** Curva de aprendizaje empinada, complejo.

### 2.1.3 Vue.js

- ✚ **Descripción:** Framework progresivo de JavaScript para construir interfaces de usuario.
- ✚ **Características:**
  - Integración gradual.
  - Reactividad.
  - Componentes.
- ✚ **Ventajas:** Fácil de aprender, flexible, buena documentación.
- ✚ **Desventajas:** Menos recursos y herramientas en comparación con React y Angular.



#### 2.1.4 Svelte

✚ **Descripción:** Framework para construir interfaces de usuario que compila eficientemente.

✚ **Características:**

- Compilación en tiempo de desarrollo.
- Sin virtual DOM.
- Código más simple.

✚ **Ventajas:** Rendimiento muy alto, menos código.

✚ **Desventajas:** Comunidad más pequeña, no está tan desarrollada como otras opciones.

#### 2.1.5 Ember.js

✚ **Descripción:** Framework de JavaScript para construir aplicaciones web ambiciosas, similar a Vue.js pero con más dificultad y complejidad.

✚ **Características:**

- Convención sobre configuración.
- Herramientas de desarrollo integradas.
- Enrutamiento robusto.

✚ **Ventajas:** Productividad alta, estructura sólida.

✚ **Desventajas:** Curva de aprendizaje empinada, menos flexible.

#### 2.1.6 Backbone.js

✚ **Descripción:** Biblioteca de JavaScript para estructurar aplicaciones web con modelos y vistas.

✚ **Características:**

- Ligero.
- MVP (Model-View-Presenter).
- Sincronización con servidor RESTful.



✚ **Ventajas:** Ligero, flexible.

✚ **Desventajas:** Requiere más configuración, menos soporte oficial.

### 2.1.7 Preact

✚ **Descripción:** Alternativa ligera a React.

✚ **Características:**

- API similar a React.
- Tamaño muy pequeño y eficiente.
- Virtual DOM.

✚ **Ventajas:** Muy ligero y un gran rendimiento.

✚ **Desventajas:** Menos características integradas, menos comunidad y menos información.

## 2.2 Tecnologías del BackEnd

### 2.2.1 Node.js

✚ **Descripción:** Entorno de ejecución para JavaScript del lado del servidor.

✚ **Características:**

- Event-driven.
- Non-blocking I/O.
- Gran ecosistema (NPM).

✚ **Ventajas:** Trabajo de forma asíncrona, gran comunidad.

✚ **Desventajas:** No adecuado para tareas intensivas en CPU.

### 2.2.2 Django

✚ **Descripción:** Framework web de alto nivel en Python.

✚ **Características:**

- Estructura MVC.
- ORM incorporado.
- Herramientas de administración.

✚ **Ventajas:** Seguridad, desarrollo rápido y API REST integrado.

✚ **Desventajas:** Pesado, menos flexible.

### 2.2.3 Ruby on Rails

✚ **Descripción:** Framework web en Ruby.

✚ **Características:**

- Convención sobre configuración.
- Herramientas integradas.
- ORM (ActiveRecord).

✚ **Ventajas:** Productividad, orientado a prototipos rápidos.

✚ **Desventajas:** Rendimiento, curva de aprendizaje elevada.

### 2.2.4 Spring boot

✚ **Descripción:** Framework para aplicaciones Java.

✚ **Características:**

- Microservicios.
- Configuración automática.
- Seguridad incorporada.

✚ **Ventajas:** Potente, escalable.

✚ **Desventajas:** Configuración compleja, pesado.

### 2.2.5 Express.js

✚ **Descripción:** Similar a Node.js, minimalista y ligero.


✚ **Características:**

- Middleware.
- Ruteo sencillo.
- Gran comunidad.

✚ **Ventajas:** Ligero, flexible.

✚ **Desventajas:** Requiere más configuración, menos estructurado.


### 2.2.6 Flask

 **Descripción:** Microframework web en Python, similar a Django pero menos complejo.


 **Características:**

- Minimalista.
- Extensible.
- WSGI.

 **Ventajas:** Ligero, fácil de aprender.


 **Desventajas:** No tiene muchas herramientas integradas.

### 2.2.7 ASP.NET Core

 **Descripción:** Framework para construir aplicaciones web modernas en .NET.

 **Características:**

- Cross-platform.
- Inyección de dependencias.
- Buen rendimiento.

 **Ventajas:** Integración con el ecosistema .NET además de un buen rendimiento.

 **Desventajas:** Curva de aprendizaje y más complejo para principiantes.

## 2.3 Selección de tecnologías

Las tecnologías de FrontEnd y BackEnd que se han analizado representan lo último en herramientas y frameworks utilizados por empresas para desarrollar aplicaciones web modernas. La elección de unas sobre otras depende de varios factores como pueden ser: requisitos del proyecto o necesidades de escalabilidad y rendimiento.

### 2.3.1 FrontEnd

En nuestro caso se ha elegido utilizar Angular en el apartado de FrontEnd por las siguientes características:

#### **1. Desarrollo de SPAs (Single Page Applications)**

Angular es ideal para construir aplicaciones de una sola página (SPA) que proporcionan una experiencia de usuario fluida y rápida sin necesidad de recargar la página.

#### **2. Estructura Modular**

Utiliza una arquitectura basada en módulos que permite la organización del código, facilitando mantenimiento y escalabilidad del proyecto.

#### **3. TypeScript**

Angular está construido sobre TypeScript, similar a JavaScript con tipos estáticos y observables. Esto mejora la calidad del código y facilita la detección de errores durante el desarrollo.

#### **4. Inyección de dependencias**

Angular tiene un sistema de inyección de dependencias que mejora el modularidad y facilita la reutilización de componentes.

#### **5. CLI (Command Line Interface)**

Angular CLI facilita la creación de nuevos proyectos, la generación de componentes, servicios, y más, además de simplificar diferentes tareas como pruebas o despliegues.

#### **6. Soporte y comunidad**

Angular es desarrollado y mantenido por Google, lo que garantiza su continuidad y evolución. Además, cuenta con una gran comunidad de desarrolladores y recursos de aprendizaje.

### 2.3.2 BackEnd

Para la parte del BackEnd se ha decidido utilizar ASP.NET Core dadas las siguientes características:

#### **1. Rendimiento**

ASP.NET Core está diseñado para ser un framework de alto rendimiento. Su arquitectura modular y ligera contribuye a mejorar la velocidad y eficiencia de las aplicaciones.

#### **2. Cross-platform**

A diferencia de su predecesor, ASP.NET Core es completamente multiplataforma, permitiendo que las aplicaciones se ejecuten en Windows, macOS y Linux, aunque en nuestro caso, no es relevante dado que vamos a hacer la aplicación solo para Windows.

#### **3. Inyección de dependencias integrada**

ASP.NET Core tiene un sistema de inyección de dependencias incorporado que facilita la gestión de dependencias (paquetes Nuggets).

#### **4. Modularidad y flexibilidad**

ASP.NET Core permite agregar solo los componentes necesarios con lo que, gracias a esta característica, podemos mejorar mucho el rendimiento y ligereza del proyecto.

#### **5. Compatibilidad con Microservicios**

Es ideal para arquitecturas de microservicios, permitiendo la construcción de aplicaciones escalables.

#### **6. Seguridad**

ASP.NET Core proporciona características de seguridad robustas, como protección contra ataques CSRF.

## 7. Ecosistema .NET

Se integra perfectamente con el ecosistema .NET, permitiendo el uso de bibliotecas y herramientas existentes y aprovechando las capacidades de C#.

Además de todas las características mencionadas anteriormente, otras de las cosas más importantes es que para hacer este TFG, era necesario usar tecnologías que ayudaran personalmente a la incorporación al mundo laboral. Son tecnologías de las más usadas y nuevas, ambas tienen un gran rendimiento y un soporte a largo plazo, por lo que son ideales para la realización de este proyecto.

## 3. ALCANCE DEL PROYECTO

Antes de comenzar con la parte práctica del trabajo, es conveniente hacer un estudio del propio desarrollo del producto, para lograrlo se ha hecho uso de tres prácticas estándar extendidas en la industria del desarrollo de software, la elicitación de requisitos, la evaluación de riesgos y un pequeño análisis de viabilidad, que sirve para proporcionar un poco más de información sobre qué usuarios podrían llegar a hacer uso de la aplicación.

### 3.1 Elicitación de requisitos del producto

#### 3.1.1 Requisitos de información

ID	Descripción	Prioridad	Criterio de aceptación
RI-01	Debe almacenar los siguientes datos: nombre, apellidos, nombre de usuario, contraseña, avatar ,dirección de correo electrónico e IBAN	Crítica	La base de datos debe permitir almacenar los datos mencionados y consultarlos
RI-02	Debe permitir almacenar, descripción, cantidad, persona que hace, persona que recibe y fecha sobre una transferencia	Critica	Todas las transferencias deben tener estos datos

RI-03	Debe almacenar un registro de todas las transferencias del usuario	Alta	Debe poder revisar en todo momento sus transferencias, así como sus movimientos
RI-04	Debe almacenar datos de ingresos y extracciones, de dinero a la cuenta del usuario, con su fecha y cantidad correspondiente	Alta	Debe poder revisar en todo momento los ingresos y extracciones de la cuenta del usuario

TABLA 1. REQUISITOS DE INFORMACIÓN

### 3.1.2 Requisitos funcionales

A continuación, veremos los requisitos funcionales de nuestra aplicación.

ID	Descripción	Prioridad	Criterio de aceptación
RF-01	Debe permitir un inicio de sesión para el acceso a los datos del usuario	Crítica	Debe controlarse el acceso a los datos mediante la sesión activa
RF-02	Debe permitir la modificación o eliminación de los datos de la cuenta de un usuario	Critica	El usuario debe tener control sobre los datos de su cuenta
RF-03	Debe permitir al usuario hacer ingresos o extracciones en la cuenta	Alta	Debe existir un formulario para llevar a cabo estas acciones
RF-04	Debe posibilitar hacer transferencias de una cuenta a otra registrada en la aplicación	Alta	Las transferencias tendrán un máximo y un mínimo de dinero, nunca podrá ser superior al dinero íntegro en la cuenta
RF-05	Debe tener una lista de movimientos para la revisión de estos por parte del usuario	Alta	Se hará una lista paginada la cual sea agradable y sencilla
RF-06	Debe poder revisar en detalle los datos de cualquier movimiento en la cuenta	Alta	El usuario puede ver en cualquier momento los datos de los movimientos de su cuenta



RF-07	Debe mostrar información de contacto del administrador de la aplicación	Baja	Ha de existir un apartado que cuente con la dirección de correo electrónico del administrador
RF-08	El administrador debe poder modificar y eliminar cualquier dato de la aplicación	Media	El administrador tiene total acceso y poder sobre la aplicación
RF-09	Debe poder exportar los datos de los movimientos de una cuenta en formato PDF	Media	El usuario puede exportar los datos de los movimientos de la cuenta

TABLA 2. REQUISITOS FUNCIONALES

### 3.1.3 Requisitos no funcionales

#### 3.1.3.1 Requisitos de calidad

En este apartado se verán los requisitos de calidad.

ID	Descripción	Prioridad	Criterio de aceptación
RC-01	La disponibilidad del sistema será la más alta posible	Alta	El sistema deberá ser estable, sin problemas y cierres inesperados
RC-02	El sistema deberá ser rápido y eficiente para una mayor facilidad del uso	Media	El sistema será rápido para un mejor uso de este

TABLA 3. REQUISITOS DE CALIDAD

#### 3.1.3.2 Requisitos de implementación

En este apartado se expondrán los requisitos de implementación.

ID	Descripción	Prioridad	Criterio de aceptación
RIM-01	Se implementara en una versión de escritorio para escritorio (Windows)	Alta	Será desarrollada en Angular y ASP .NET Core
RIM-02	Empleará una base de datos para almacenar los datos de la aplicación	Media	Será una base de datos relacional y contendrá todos los datos de la aplicación

TABLA 4. REQUISITOS DE IMPLEMENTACIÓN

### 3.1.3.3 Requisitos de seguridad

En este apartado veremos los requisitos de seguridad necesarios en la aplicación.

ID	Descripción	Prioridad	Criterio de aceptación
RS-01	La aplicación no tendrá ningún código ejecutable salvando librerías de terceros	Media	Aunque es código libre, el código oficial de la aplicación ha de ser protegido
RS-02	No se podrá modificar movimientos por parte del usuario, solo por parte del administrador y en caso de ser necesario	Alta	La aplicación mantendrá la integridad de los datos del usuario
RS-03	Debe garantizar la confidencialidad, integridad y autenticidad de la información transmitida	Crítica	Especialmente los datos del usuario estarán protegidos de forma robusta y no saldrán de la aplicación
RS-04	No se debe ejecutar operaciones con privilegios en modo súper usuario	Alta	No se debe solicitar el modo administrador (root) en ningún momento
RS-05	Los datos sensibles han de estar encriptados	Alta	Datos como contraseñas estarán encriptados con un algoritmo fiable
RS-06	Se validarán los formularios para evitar inyecciones de código malicioso	Crítica	Todos los formularios contarán con validación
RS-07	La aplicación no contendrá ningún software malicioso o perjudicial para el dispositivo del usuario	Alta	La aplicación no contendrá ningún software malicioso, con fines lucrativos o dañinos

### 3.2 Riesgos

La clasificación de los riesgos que pueden afectar a la aplicación se estimará multiplicando la probabilidad por el impacto del riesgo. Para determinar el impacto de un riesgo se tienen en cuenta tres parámetros, la modificación de alcance de la aplicación, repercusión en la calidad y necesidad de revisiones de la planificación. Para realizar la estimación se ha utilizado la siguiente figura:

Matriz de Probabilidad e Impacto		Impacto				
		Muy bajo 0.05	Bajo 0.1	Moderado 0.2	Alto 0.4	Muy Alto 0.8
Probabilidad	Muy alta 0.9	0.045	0.09	0.18	0.36	0.72
	Alta 0.7	0.035	0.07	0.14	0.28	0.56
	Moderada 0.5	0.025	0.05	0.1	0.2	0.4
	Baja 0.3	0.015	0.03	0.06	0.12	0.24
	Muy Baja 0.1	0.005	0.01	0.02	0.04	0.08

Riesgo Bajo ---- Riesgo Moderado ---- Riesgo Alto

#### ILUSTRACIÓN 1. MATRIZ DE PROBABILIDAD E IMPACTO

Donde la probabilidad e impacto máxima es 0.72.

Para su evaluación, primero se consideran los riesgos entre 10% y 90% de probabilidades de que ocurra, luego se decidirá el impacto de ese riesgo, y se hará una multiplicación entre la columna y la fila correspondiente para obtener la puntuación.

ID	Descripción	Probabilidad	Impacto	Puntuación	Riesgo
1	Estimación incorrecta de la planificación	Moderada	Muy Alto	0.4	Alto
2	Subestimar el alcance	Alta	Alto	0.28	Alto
3	Cambio de requisitos de avance del proyecto	Media	Alto	0.2	Medio

4	Incompatibilidad en funcionalidades tecnológicas	Baja	Bajo	0.03	Bajo
5	Alta complejidad en el desarrollo del modelo	Medio	Alto	0.2	Medio
6	Bajo rendimiento en la aplicación	Bajo	Medio	0.06	Medio
7	Baja aceptación de la aplicación	Bajo	Bajo	0.03	Bajo

TABLA 5. TABLA DE RIESGOS

Como podemos ver en la tabla 5, hay varios riesgos a considerar en el desarrollo de la aplicación, aunque vemos que la mayoría de ellos suponen un riesgo bajo/medio, hay algunos riesgos altos, aunque sean en la parte organizativa de la aplicación.

### 3.3 Viabilidad

En este caso hemos hecho una encuesta anónima en la cual hay 6 preguntas hechas mediante la plataforma de google forms, aunque también hay una persona que es profesor de economía que pretende usar la aplicación en sus clases.

### 3.4 Objetivos de la aplicación

El objetivo principal del trabajo es la investigación y desarrollo de una aplicación web orientada a ser un ejemplo general de un banco el cual tiene todas las funcionalidades cores de este, para que los usuarios, puedan practicar la utilización de aplicaciones del mismo estilo y haya una mayor comprensión general de estas.

## 4. VERSIÓN DE COSTES

### 4.1 Coste salarial del desarrollo

En este apartado se desglosan los costes salariales del equipo de desarrollo que contempla desde la primera semana de vida del proyecto hasta su puesta en producción, en este caso, 15 semanas. Cabe destacar de que se va a trabajar unas 20 horas semanales en el proyecto.

Rol	Sueldo por hora		Sueldo por semana		Coste Total mensual	
	Bruto	Neto	Bruto	Neto	Bruto	Neto
Project Manager	23,50€	16,45€	470€	329€	1880€	1316€

**TABLA 6. COSTE DE DESARROLLO DEL EQUIPO**

Como solo es una persona la que realiza el proyecto, el coste salarial del equipo mensual es igual al coste mensual del integrante.

Evento	Duración (semanas)	Coste total	
		Bruto	Neto
Preparación del proyecto	4	1880€	1316€
Sprint 1	3	1410€	987€
Sprint 2	3	1410€	987€
Sprint 3	3	1410€	987€
Preparación	2	940€	658€
<b>Total</b>	<b>15</b>	<b>7.050€</b>	<b>4.935€</b>

**TABLA 7. COSTE TOTAL SALARIAL DEL DESARROLLO DEL PROYECTO**

### 4.2 Costes derivados del desarrollo

Como su propio nombre indica, aquí se describirán y definirán los costes derivados del desarrollo, en el mismo periodo de tiempo que hemos definido hasta la puesta en producción, 15 semanas.

Tipo de gastos	Gasto semanal	Gasto mensual
Gasto eléctrico	15€	60€
Gasto de internet	15€	60€

TABLA 8. COSTES DERIVADOS EN EL DESARROLLO

Estos gastos se deben a la compensación de que sea un trabajo remoto.

Evento	Duración (semanas)	Coste total
Preparación del proyecto	4	60€
Sprint 1	3	45€
Sprint 2	3	45€
Sprint 3	3	45€
Preparación	2	30€
<b>Total</b>	<b>15</b>	<b>225€</b>

TABLA 9. COSTES DERIVADOS EN EL DESARROLLO TOTALES

#### 4.3 Costes de licencias

A continuación, trataremos el desglose de los costes asociados a las licencias usadas durante el periodo de desarrollo. Teniendo en cuenta los precios sacados de la [página oficial de Microsoft](#), y obteniendo el Microsoft 365 E3 EEA (sin teams).

Licencia	Gasto por uso mensual
Microsoft Office	35,70€
ASP .NET Core	0€
Angular	0€
<b>Total</b>	<b>35,70€</b>

TABLA 10. COSTE DE LICENCIAS MENSUAL

Para el desarrollo y conteo de horas también usaremos Clockify, el cual también es gratuito y no necesitaremos pagar la licencia.

Evento	Duración (semanas)	Coste total
Preparación del proyecto	4	35,70€
Sprint 1	3	26,80€
Sprint 2	3	26,80€
Sprint 3	3	26,80€
Preparación	2	17,85€
<b>Total</b>	<b>15</b>	<b>133,95€</b>

TABLA 11. COSTES DE LICENCIAS TOTALES

#### 4.4 Coste del material de trabajo

Aquí se describen los gastos de coste de material de trabajo adquiridos durante el periodo de desarrollo, debido a que su vida útil será de más de 15 semanas de vida, se considerará las amortizaciones de los equipos y materiales adquiridos durante el periodo de desarrollo durante un año de vida.

Item	Precio/unidad
Ordenador portátil	600€

TABLA 12. PRECIO MATERIAL DE TRABAJO

Amortización de los equipos	
Suma total	Periodo de amortización
600	1 año
Amortización semanal	Amortización mensual
12,50€	50€

TABLA 13. AMORTIZACIÓN DE LOS EQUIPOS

Amortización durante el desarrollo	
Periodo de desarrollo	15 semanas
Amortización semanal	12,50€
<b>Total</b>	<b>187,50€</b>

TABLA 14. AMORTIZACIÓN DURANTE EL DESARROLLO

Teniendo en cuenta los cálculos finales, en casi 4 meses se va a amortizar unos 187,50€, todo con el I.V.A incluido.



#### 4.5 Desglose total de los costes de desarrollo

En este subapartado están resumidos todos los costes asociados al periodo de desarrollo del producto.

Costes totales del desarrollo (sin reservas de contingencia)	
Costes salariales totales	4935€
Costes derivados totales	225€
Costes de licencia totales	133,95€
Costes de material total	600€
<b>Total</b>	<b>5893,95€</b>

**TABLA 15. COSTES TOTALES DEL DESARROLLO**

Si le sumamos la reserva de contingencia del siguiente apartado se nos quedaría en **6.483,35€**.

#### 4.6 Reservas de contingencia

Aquí se definirán las reservas de contingencia establecidas para el proyecto, que serán de un 10% del coste total (sin contar la propia reserva).

Reservas de contingencia	
Porcentaje de contingencia	10%
Presupuesto total	5893,95€
<b>Total</b>	<b>589,40€</b>

**TABLA 16. RESERVAS DE CONTINGENCIA**