Documentazione NebulaWatches

1 Introduzione 3

1.1 Informazioni sul progetto 3

1.2 Abstract 3

1.3 Scopo 3

2 Analisi 4

2.1 Analisi del dominio 4

2.2 Analisi e specifica dei requisiti 4

2.3 Use case 7

2.4 Pianificazione 7

2.5 Analisi dei mezzi 8

2.5.1 Software 8

2.5.2 Hardware 8

3 Progettazione 9

3.1 Design dell’architettura del sistema 9

3.2 Design dei dati e database 9

3.3 Design delle interfacce 9

3.4 Design procedurale 9

4 Implementazione 10

5 Test 10

5.1 Protocollo di test 10

5.2 Risultati test 11

5.3 Mancanze/limitazioni conosciute 11

6 Consuntivo 11

7 Conclusioni 11

7.1 Sviluppi futuri 11

7.2 Considerazioni personali 11

8 Glossario 11

9 Bibliografia 12

9.1 Bibliografia per articoli di riviste: 12

9.2 Bibliografia per libri 12

9.3 Sitografia 12

10 Allegati 12

# Introduzione

## Informazioni sul progetto

* Titolo Progetto: NebulaWatches
* Allievo: Alexandru Ciobanu I3BB, Gioele Chiodoni I3BB, Tom Schillerwein I3BC
* Docente responsabile: Guido Montalbetti
* Scuola: Arti e Mestieri Trevano, sezione Informatica
* Data di inizio e fine: 12.01.2024 – 03.05.2024
* Data di presentazione: 17 – 24 .05.2024

## Abstract

For this project, a web application needs to be developed to manage a watch store.

Users have several basic management tools at their disposal, such as a list of watches from which they can compose their inventory, do advanced watch searches, and view their customers and what they buy.

They also have at their disposal advanced tools such as a chatbot, based on data from out Database, and a company message centralization point to improve and optimize the management of their watch store.

In addition, this project is about learning how to manage a project in the agile mode and how to work and share work in a team.

## Scopo

Per questo progetto bisogna sviluppare un’applicazione web che permetta la gestione di un negozio di orologi.

Gli utenti hanno a disposizione diversi tool di gestione base, come una lista di orologi da cui possono comporre il loro inventario, fare ricerche avanzate sugli orologi, visualizzare i propri clienti e ciò che comprano.

Inoltre hanno a disposizione dei tool avanzati come un chatbot, basato sui dati del database, e un punto di centralizzazione dei messaggi dell’azienda per migliorare e ottimizzare la gestione del proprio negozio di orologi.

Inoltre questo progetto serve a imparare a gestire un progetto nella modalità agile e a lavorare e spartirsi il lavoro in un team.

# Analisi

## Analisi del dominio

Abbiamo scelto di sviluppare un applicativo web che permetta all’utente di migliorare e semplificare la gestione del proprio negozio di orologi. Il prodotto deve essere facile da usare e presentare un’interfaccia grafica moderna. A nostro sapere non esiste un prodotto specifico per la gestione dei negozi di orologi sul mercato.

## Analisi e specifica dei requisiti

|  |  |
| --- | --- |
| **ID: REQ-001** | |
| **Nome** | Accesso e login |
| **Priorità** | 1 |
| **Versione** | 1.0 |
| **Note** | Un utente non registrato può solo vedere la lista degli orologi, un utente registrato può sfruttare tutte le funzionalità del sito. |
| **Sotto requisiti** | |
| **001** | Maschera di login |

|  |  |
| --- | --- |
| **ID: REQ-002** | |
| **Nome** | Registrazione |
| **Priorità** | 1 |
| **Versione** | 1.0 |
| **Note** | Un utente senza account deve poter registrarsi tramite email oppure tramite Google |
| **Sotto requisiti** | |
| **001** | Maschera di registrazione |
| **002** | Se con email bisogna fare la verifica dell’email |

|  |  |
| --- | --- |
| **ID: REQ-003** | |
| **Nome** | Database |
| **Priorità** | 1 |
| **Versione** | 1.0 |
| **Note** | Database contenente orologi, utenti, inventario dell’utente, clienti dell’utente e team dell’utente |

|  |  |
| --- | --- |
| **ID: REQ-004** | |
| **Nome** | Chatbot |
| **Priorità** | 1 |
| **Versione** | 1.0 |
| **Note** | Chatbot basato su dati del database che aiuta e dà consigli all’utente |

|  |  |
| --- | --- |
| **ID: REQ-005** | |
| **Nome** | Comunication center |
| **Priorità** | 1 |
| **Versione** | 1.0 |
| **Note** | Centro di raggruppamento email |

|  |  |
| --- | --- |
| **ID: REQ-006** | |
| **Nome** | Orologi nel magazzino |
| **Priorità** | 1 |
| **Versione** | 1.0 |
| **Note** | L’utente può selezionare gli orologi che ha nel proprio inventario/magazzino |
| **Sotto requisiti** | |
| **001** | Orologi venduti e comprati |
| **002** | Informazioni su spedizione, consegna, ecc. |

|  |  |
| --- | --- |
| **ID: REQ-007** | |
| **Nome** | Orologi |
| **Priorità** | 1 |
| **Versione** | 1.0 |
| **Note** | Gli orologi sono ordinati per brand. |
| **Sotto requisiti** | |
| **001** | Ricerca e filtri di ricerca |
| **002** | Preferiti |

|  |  |
| --- | --- |
| **ID: REQ-008** | |
| **Nome** | Team |
| **Priorità** | 1 |
| **Versione** | 1.0 |
| **Note** | L’utente può aggiungere e gestire i propri membri del team |
| **Sotto requisiti** | |
| **001** | Informazioni di vendita dei membri del team |

|  |  |
| --- | --- |
| **ID: REQ-009** | |
| **Nome** | Clienti |
| **Priorità** | 1 |
| **Versione** | 1.0 |
| **Note** | Informazioni sui propri clienti |
| **Sotto requisiti** | |
| **001** | Orologi comprati |

|  |  |
| --- | --- |
| **ID: REQ-010** | |
| **Nome** | Admin |
| **Priorità** | 1 |
| **Versione** | 1.0 |
| **Note** | Gestione CRUD degli utenti |
| **Sotto requisiti** | |
| **001** | Paginazione e ricerca dei utenti. |

## Use case

|  |
| --- |
| Diagramma di Gantt Preventivo |

## Pianificazione

Prima di stabilire una pianificazione bisogna avere almeno una vaga idea del modello di sviluppo che si intende adottare. In questa sezione bisognerà inserire il modello concettuale di sviluppo che si seguirà durante il progetto. Gli elementi di riferimento per una buona pianificazione derivano da una scomposizione top-down della problematica del progetto.

La pianificazione può essere rappresentata mediante un diagramma di Gantt:

|  |
| --- |
| Diagramma di Gantt Preventivo |

Se si usano altri metodi di pianificazione (p.es. scrum), dovranno apparire in questo capitolo.

## Analisi dei mezzi

Per questo progetto usiamo 3 PC scolastici su cui eseguiamo delle macchine virtuali Linux Mint per il development.

### Software

* Visual Studio Code 1.85.2, come editor di testo
* IntelliJ IDEA Community Edition 2023.3.2, come IDE
* Google Chrome 116.0, come browser per visualizzare il sito durante lo sviluppo
* Mozilla Firefox 110.0, per testare la visualizzazione del sito
* teamgantt.com, per fare il Gantt
* Microsoft Visio, per fare lo Use Case
* Microsoft Word e Obsidian, per redigere la documentazione, il QdC e il diario
* GitHub, come Repository online
* Postman v10.22 per testare gli endpoint dell’API.

### Hardware

Computer utilizzato come host delle macchine virtuali:

* Processore: Intel(R) Core(TM) i7-9700 CPU @ 3.00GHz
* RAM: 32GB
* Scheda Video: NVIDIA GeForce RTX 2060
* SSD: 512GB
* 2 Display 1920x1080 60Hz

# Progettazione

Questo capitolo descrive esaustivamente come deve essere realizzato il prodotto fin nei suoi dettagli. Una buona progettazione permette all’esecutore di evitare fraintendimenti e imprecisioni nell’implementazione del prodotto.

## Design dell’architettura del sistema

Descrive:

* La struttura del programma/sistema lo schema di rete...
* Gli oggetti/moduli/componenti che lo compongono.
* I flussi di informazione in ingresso ed in uscita e le relative elaborazioni. Può utilizzare *diagrammi di flusso dei dati* (DFD).
* Eventuale sitemap

## Design dei dati e database

Descrizione delle strutture di dati utilizzate dal programma in base agli attributi e le relazioni degli oggetti in uso.

Schema E-R, schema logico e descrizione.

Se il diagramma E-R viene modificato, sulla doc dovrà apparire l’ultima versione, mentre le vecchie saranno sui diari.

## Design delle interfacce

Descrizione delle interfacce interne ed esterne del sistema e dell’interfaccia utente. La progettazione delle interfacce è basata sulle informazioni ricavate durante la fase di analisi e realizzata tramite mockups.

## Design procedurale

Descrive i concetti dettagliati dell’architettura/sviluppo utilizzando ad esempio:

* Diagrammi di flusso e Nassi.
* Tabelle.
* Classi e metodi.
* Tabelle di routing
* Diritti di accesso a condivisioni …

Questi documenti permetteranno di rappresentare i dettagli procedurali per la realizzazione del prodotto.

# Implementazione

## Login

Nel nostro applicativo il login si può fare con due modalità. La prima tipologia di login è basata su JWT (JSON web token), la seconda è un login con Google.

### Login JWT

Per registrare un nuovo utente, l’applicazione necessita di un email, username e password. Se la richiesta è valida, l’applicativo ritorna un token che serve a dire al server che questo utente è autenticato.

Per autenticare un utente già esistente, l’applicazione necessita di email e password dell’utente. Anche in questo caso, se la richiesta è valida, ritorna un token che segnala al server che questo utente è autenticato.

L’utente invia il JWT ricevuto in ogni richiesta successiva e l’applicazione verifica il JWT per autenticare l’utente.

Se il token sta per scadere, l’applicazione può decidere di rinnovarlo e inviare un nuovo JWT all’utente.

### Login Google

Quando un utente vuole accedere, l’applicazione lo reindirizza alla pagina di login di Google. L’utente inserisce poi le proprie credenziali di Google e viene autenticato se questo è andato a buon fine.

# Test

## Protocollo di test

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case:**  **Riferimento**: | TC-001  REQ-001 | **Nome:** | Endpoint registrazione utente |
| **Descrizione:** | Passando username, email, password, viene creato l’utente e viene generato un token di autenticazione. | | |
| **Prerequisiti:** | Installato Postman, bisogna avere il server MySQL, avere il server Spring acceso. | | |
| **Procedura:** | 1. Accendere Postman 2. Creare collezione “NebulaWatches” 3. Creare richiesta POST e mettere come URL “localhost:64321/api/v1/auth/register” poi andare nella tab “Body/raw”: { “username” : “gino”, “email”: “gino@gmail.com”, “password”: “1234” } 4. Schiacciare Send | | |
| **Risultati attesi:** | Nella response un token di autorizazzione | | |

## Risultati test

## Mancanze/limitazioni conosciute

Descrizione con motivazione di eventuali elementi mancanti o non completamente implementati, al di fuori dei test case. Non devono essere riportati gli errori e i problemi riscontrati e poi risolti durante il progetto.

# Consuntivo

Consuntivo del tempo di lavoro effettivo e considerazioni riguardo le differenze rispetto alla pianificazione (cap. 1.7) (ad esempio Gantt consuntivo).

# Conclusioni

Quali sono le implicazioni della mia soluzione? Che impatto avrà? Cambierà il mondo? È un successo importante? È solo un’aggiunta marginale o è semplicemente servita per scoprire che questo percorso è stato una perdita di tempo? I risultati ottenuti sono generali, facilmente generalizzabili o sono specifici di un caso particolare? ecc.

## Sviluppi futuri

Migliorie o estensioni che possono essere sviluppate sul prodotto.

## Considerazioni personali

Cosa ho imparato in questo progetto? ecc.

# Glossario

Inserite una semplice tabella con due colonne che spieghi i termini specifici del progetto (lista dei termini in ordine alfabetico A-Z)

Esempio:

|  |  |
| --- | --- |
| **Termine** | **Descrizione** |
| URL | **Uniform Resource Locator**: è una stringa di caratteri che identifica e individua una risorsa su Internet |
|  | . |

# Bibliografia

## Bibliografia per articoli di riviste:

1. Cognome e nome (o iniziali) dell’autore o degli autori, o nome dell’organizzazione,
2. Titolo dell’articolo (tra virgolette),
3. Titolo della rivista (in italico),
4. Anno e numero
5. Pagina iniziale dell’articolo.

## Bibliografia per libri

1. Cognome e nome (o iniziali) dell’autore o degli autori, o nome dell’organizzazione,
2. Titolo del libro (in italico),
3. ev. Numero di edizione,
4. Nome dell’editore,
5. Anno di pubblicazione,
6. ISBN.

## Sitografia

1. URL del sito (se troppo lungo solo dominio, evt completo nel diario),
2. Eventuale titolo della pagina (in italico),
3. Data di consultazione (GG-MM-AAAA).

**Esempio:**

* http://standards.ieee.org/guides/style/section7.html, *IEEE Standards Style Manual*, 07-06-2008.

# Allegati

Elenco degli allegati, esempio:

* Diari di lavoro
* Codici sorgente/documentazione macchine virtuali
* Istruzioni di installazione del prodotto (con credenziali di accesso) e/o di eventuali prodotti terzi
* Documentazione di prodotti di terzi
* Eventuali guide utente / Manuali di utilizzo
* Mandato e/o QdC
* Prodotto
* …