Titolo del documento

Autori

Data di consegna

Sommario

[Analisi](#_heading=h.gjdgxs) **3**

[Esempio di paragrafo](#_heading=h.30j0zll) 3

[Requisiti funzionali](#_heading=h.1fob9te) 3

[Caso d’uso](#_heading=h.3znysh7) 3

[Progettazione](#_heading=h.2et92p0) **5**

[Tipi di dato e strutture dati](#_heading=h.tyjcwt) 5

[Librerie e funzioni](#_heading=h.3dy6vkm) 5

[Dipendenze tra funzioni](#_heading=h.1t3h5sf) 5

[Flow chart/pseudo-codice](#_heading=h.17dp8vu) 5

Codifica **6**

[Test](#_heading=h.26in1rg) **7**

[Conclusioni](#_heading=h.lnxbz9) **8**

Analisi

Descrivere, a un alto livello di astrazione, i principali aspetti legati al problema che si vuole affrontare. Descrivere, in modo discorsivo, le funzionalità, i potenziali utenti, le caratteristiche principali, ecc.

Strutturare il testo in capoversi, con andate a capo, per migliorare la leggibilità.

Strutturare la trattazione con paragrafi e/o sotto-paragrafi.

Esempio di paragrafo

Semplificare la trattazione con elenchi puntati o numerati, con immagini e tabelle.

Esempio di elenco puntato:

* Punto 1;
* Punto 2;
* Punto 3.

Ogni immagine o tabella deve avere una didascalia e deve essere descritta nel testo o perlomeno menzionata, altrimenti è inessenziale (vedi Tabella 1).

| Etichetta di colonna | Etichetta di colonna | Etichetta di colonna |
| --- | --- | --- |
|  |  |  |
|  |  |  |

**Tabella 1.** Esempio di tabella.

Requisiti funzionali

Descrivere schematicamente le funzionalità implementate. Assegnare un codice univoco a ciascuna funzionalità e fornire una descrizione.

| Codice | Nome | Descrizione |
| --- | --- | --- |
| R01 | Visualizzazione menù | Il programma deve mostrare all’utente un menù iniziale con le opzioni disponibili. |
| R02 | … | … |
| … | … | … |

**Tabella 2.** Requisiti funzionali.

Casi d’uso

Descrivere i casi d’uso relativi a ciascun requisito del sistema. Un requisito può essere associato a più di un caso d’uso. Descrivere: pre-condizioni, che devono verificarsi per poter utilizzare quella particolare funzionalità, post-condizioni, evento innescante, cioè come si arriva a quel punto del programma, scenario di base e scenario alternativo.

| Codice | Nome | Descrizione |
| --- | --- | --- |
| R09 | Visualizzazione profilo | Il programma deve mostrare le informazioni sugli acquisti effettuati dall’utente. |
| Pre-condizioni | Il codice dell’utente dev’essere valido. L’utente deve aver espresso almeno una preferenza. | |
| Post-condizioni | Successo | Il sistema visualizza il profilo. |
|  | Fallimento | Il sistema mostra un messaggio di errore. |
| Scenario di base | Il sistema visualizza il profilo utente. (Se lo scenario di base porta a un altro caso d’uso, indicarne il codice.) | |
| Scenario alternativo | Il sistema mostra un messaggio di errore. (Se lo scenario alternativo porta a un altro caso d’uso, indicarne il codice.) | |

**Tabella 3.** Caso d’uso del requisito R09.

Progettazione

Descrivere, a un livello di astrazione intermedio, gli aspetti principali della soluzione proposta.

Principali variabili, strutture dati e file

Indicare i tipi di dato e le strutture dati utilizzati nel caso di studio. Indicare anche eventuali file utilizzati e il loro scopo.

| Nome | Tipologia | Descrizione | Tipi/campi/valori |
| --- | --- | --- | --- |
| user | struct | Tipo di dato definito per descrivere le caratteristiche dell’utente. | Nome: char[20]  Cognome: char[20]  … |
| preferences | enum | Tipo di dato per memorizzare le scelte dell’utente. | Like/dislike |
| A | Variabile globale | Variabile utilizzata per… | … |
| data.csv | File | File utilizzato per conservare… | … |

**Tabella X.** Tipi di dato e strutture dati.

Librerie e funzioni

Indicare quali librerie sono state progettate. Per ciascun file .h indicare le procedure e le funzioni incluse nell’header. Per ogni funzione indicare scopo, tipi di ingresso e di uscita.

Dipendenze tra funzioni

Per ciascuna delle funzioni progettate, indicare le eventuali dipendenze. Per esempio, la funzione di visualizzazione del menù richiama a sua volta le funzioni del programma.

Flow chart/pseudo-codice

Per ciascuna delle funzioni progettate, utilizzare flow-chart o pseudo-codice per schematizzarne l’implementazione.

Codifica

Descrivere, a un basso livello di astrazione, gli aspetti pragmatici della soluzione proposta.

Aiutarsi con frammenti di codice degli algoritmi o delle funzioni più rilevanti, descrivendoli opportunamente nel testo.

Eventualmente, allegare la documentazione prodotta con Doxygen.

Test

Mostrare esempi di esecuzione (a un determinato input corrisponde un determinato output).

Per ciascun caso d’uso definito nel capitolo 1, definire i casi di test e validarne l’esito.

| Codice requisito | Codice test | Nome | Descrizione test | Eventuale input | Risultato atteso | Risultato ottenuto |
| --- | --- | --- | --- | --- | --- | --- |
| R01 | 1.1 | Menù iniziale | Scelta n. 1 | 1 | Login | Indicare risultato |
| R01 | 1.2 | Menù iniziale | Scelta errata | 100 | Messaggio di errore | … |
| R02 | 2.2 | Caricamento dati da file | File non esistente | / | Messaggio di errore | … |
| … | … | … | … | … | … | … |

**Tabella Y.** Risultati dei test.

Conclusioni

Tirare le somme del lavoro svolto, evidenziando i punti di forza e di debolezza della soluzione proposta e gli eventuali sviluppi futuri volti a migliorarla.

In particolare, commentare gli esiti del piano di test, individuare eventuali criticità riscontrate e pianificare azioni migliorative.