1. Analisi

1.1 Descrizione del Sistema

<Descrivere il problema generale che il sistema di prepone di risolvere. Descrivere in modo discorsivo le funzionalità, gli utenti, le caratteristiche principali. Indicare qui anche le estensioni e le funzionalità aggiuntive implementate>

Esempio

Il sistema simula una porzione delle funzionalità della piattaforma Spotify. Nello specifico, il programma dovrà occuparsi di …. Etc etc… Gli utenti del sistema saranno definiti attraverso queste caratteristiche…. Etc etc. Accanto alle funzioni di base previste dal sistema, sono state implementate anche le seguenti funzioni aggiuntive….

1.2 Requisiti Funzionali

<Descrivere Schematicamente la funzionalità implementate. Assegnare un codice univoco a ogni funzionalità e fornire una descrizione>

Esempio

Codice	Nome	Descrizione
R01	Visualizzazione Menu	Il programma deve mostrare all'utente un menu iniziale dove vengono visualizzate le opzioni disponibili
R02	Caricamento Dati da File	Il programma deve caricare i dati da file … etc etc..
….	….	….
R09	Visualizzazione Profilo	Il programma deve mostrare le informazioni sugli

		acquisti effettuati dall'utente etc. etc.

1.3 Casi d'Uso

<Descrivere i casi d'uso collegati a ciascun requisito. Requisito può anche essere collegato più di un caso d'uso. Descrivere pre-condizioni (condizioni che devono verificarsi per poter usare quella funzionalità), evento innescante (come si arriva a quel punto del programma), Scenario di base e scenario alternativo >

Esempio

Codice	Nome	Descrizione
R09	Visualizzazione Profilo	Il programma deve mostrare le informazioni sugli acquisti effettuati dall'utente etc. etc.
Pre-Condizioni	Il codice dell'utente deve essere valido. Deve aver inserito almeno una preferenza	
Post-Condizioni	Successo	Il sistema visualizza il profilo
	Fallimento	Il sistema mostra un messaggio d'errore
Evento Innescante	L'utente deve aver selezionato la voce 'Visualizza Profilo sul menu iniziale'	
Scenario di Base	Il sistema visualizza il profilo dell'utente *(se lo scenario di base porta ad un altro caso d'uso, indicarne il codice)*	
Scenario Alternativo	Il sistema mostra un messaggio d'errore a seconda che l'utente non esista o non abbia ancora inserito delle preferenze. *(se lo scenario di base porta ad un altro caso d'uso, indicarne il codice)*	

1.4 Strumenti di Sviluppo

<Indicare gli strumenti tecnologici – Hardware e Software – utilizzati per lo sviluppo del progetto. (es. notebook, linguaggio di programmazione, IDE, etc.) Indicare eventuali documenti-materiale da cui si è attinto per approfondire le tematiche del progetto (es. delle guide). Indicare anche se ci sono dei requisiti minimi per l'esecuzione del sistema>

Esempio

Il sistema è stato realizzato utilizzando un Notebook X Y a 1.6 GHZ, con 16GB di Ram. L'ambiente di sviluppo adottato è stato Eclipse, corredato dai plugin X,Y, Z. Per implementare la funzione X ci si è avvalsi del materiale aggiuntivo disponibile qui <URL>. Per eseguire il sistema è necessario avere un computer con queste caratteristiche…blablabla

2. Progettazione

2.1 Progettazione dei tipi di dato, delle strutture dati

<Indicare i tipi di dato e le strutture dati utilizzate nell'ambito del progetto. Indicare anche eventuali file utilizzati nel programma e il relativo scopo>

Esempio

Nome	Tipologia	Descrizione	Tipi / Campi / Valori
User	struct	Tipo di dato definito per descrivere le caratteristiche dell'utente	Nome: char[20] Cognome: char[20] Etc. etc.

Preferences	enum	Tipo di Dato per memorizzare le scelte dell'utente	Like / dislike
…	…	…	…

Nome	Tipologia	Descrizione	Tipi / Campi / Valori
Max_Items	costante	Costante utilizzata per memorizzare il massimo numero di oggetti registrabili	1000
A	Variabile globale	Variabile globale utilizzata per….	int
…	…	…	…
Dat.dat	File	File utilizzato per…	….

2.2 Progettazione delle librerie / funzioni

<Indicare quali sono le librerie progettate. Per ciascun file .h indicare le procedure e le funzioni incluse nell'header file. Per ogni funzione indicare scopo, tipi di ingresso e di uscita. In caso di uso di Doxygen, limitarsi a elencare le librerie, i metodi di ciascuna libreria e il criterio di progettazione della libreria, e rimandare alla documentazione Doxygen le altre informazioni>

2.3 Dipendenza tra funzioni

<Per ognuna delle funzioni progettate, indicare le eventuali dipendenze. Ad esempio la funzione di visualizzazione del menu richiama a sua volta le

funzioni del programma. Oppure la funzione di visualizzazione del profilo (ad esempio) richiama una funzione che verifica se l'utente esiste oppure no>

2.4 Flow-Chart / Pseudo-Codice

<Per ognuna delle funzioni progettate, utilizzare i flow-chart o lo pseudo-codice per schematizzarne l'implementazione>

3. Codifica

<Allegare la Documentazione Prodotta con Doxygen>

4. Testing

4.1 Definizione del Piano di Test

<Per ciascun caso d'uso definito in sezione 1.2 definire i casi di test e validarne l'esito. Quando possibile, progettare i casi di test come asserzioni CUnit>

Codice Requisito	Codice Test	Nome	Descrizione Test	Eventuale Input	Risultato Atteso	Risultato Ottenuto
R01	1.1	Menu Iniziale	Scelta n.1	1	Caricamento del menu X	<indicare risultato>
R02	1.2	Menu iniziale	Scelta errata	100	Messaggio di Errore	<indicare risultato>
R02	2.1	Caricamento Dati da File	File non esistente	xxx	Visualizzazione Messaggio di Errore e Creazione Nuovo File	<indicare risultato>
R02	2.2	Caricamento Dati da File	File Esistente	xxx	Caricamento dei File effettuato corretamente	<indicare risultato>

4.2 Esiti del Piano di Test ed eventuali commenti

<Commentare gli esiti del piano di test, individuare eventuali criticità (test che l’attuale implementazione non riesce a superare) e pianificare eventuali azioni migliorative sul codice>