# ECORIDE

## TRACCIA 2:
## MOBILITÀ SOSTENIBILE

SAMUELE DI GRASSI
ANDREA MARTILOTTI
MARIKA SCHIAVO

13/06/2024

UNIVERSITÀ
DEGLI STUDI DI BARI
ALDO MORO

**1.Analisi** 2
1.1 Obiettivo 2
1.1.2 Utenti 2
1.1.3 Mappa 2
1.1.4 Noleggio 2
1.1.5 Guasti 2
1.1.6 Feedback 3
1.1.7 Admin 3
1.1.8 Analytics 3
1.1.9 Elenco richiesta 3
1.2 Requisiti funzionali 4
1.3 Casi d'uso 8
**2. Progettazione** **36**
2.2 Principali variabili, strutture dati e file 39
2.3 Librerie e funzioni 42
2.4 Dipendenze tra funzioni 44
2.5 Flowchart 45
**3. Codifica** **51**
3.1 API Brevo 51
3.2 Diagramma E/R 52
3.3 Gestione transazioni bici 53
3.4 Test 54
**4. Conclusioni** **71**
4.1 Punti di forza 71
4.2 Punti di debolezza 71
4.3 Potenziamenti Futuri 72
**5. Considerazioni finali e ringraziamenti** **72**

# 1.Analisi

## 1.1 Obiettivo

Una città impegnata a promuovere la sostenibilità ambientale desidera implementare un sistema di bike sharing per incoraggiare i cittadini a scegliere metodi di trasporto più ecologici, riducendo l'inquinamento atmosferico e il traffico urbano e promuovendo uno stile di vita sano.
Il sistema di bike sharing offre una serie di funzionalità sia per gli utenti che per gli amministratori del servizio.

### 1.1.2 Utenti

Gli utenti potranno creare un profilo personale fornendo i dati personali necessari al fine di gestire il proprio abbonamento e i noleggi.
All'atto della registrazione, gli utenti avranno la possibilità di scegliere tra una soluzione mensile o settimanale.

### 1.1.3 Mappa

Gli utenti potranno inoltre visualizzare una mappa che individuerà le stazioni di bike sharing più vicine, con aggiornamenti in tempo reale (sui *.csv*) relativi alla disponibilità delle biciclette, con la possibilità di seguire percorsi ciclabili consigliati.
Questo aiuterà gli utenti a pianificare i loro viaggi in modo efficiente e sicuro, incentivando l'uso delle biciclette per gli spostamenti quotidiani.

### 1.1.4 Noleggio

Gli utenti avranno la possibilità di prenotare una bicicletta in modo semplice e veloce, per un massimo di 3 ore, da ogni stazione, garantendo così per tutti gli utenti la disponibilità al momento del bisogno.

### 1.1.5 Guasti

In caso di problemi o guasti con le biciclette, gli utenti potranno segnalare rapidamente l'inconveniente tramite il sistema.
Questo canale di comunicazione indiretto permetterà una rapida risoluzione dei problemi e migliorerà l'esperienza globale degli utenti.

### 1.1.6 Feedback

Gli utenti potranno fornire feedback sulla loro esperienza, contribuendo a migliorare ulteriormente il servizio e a garantire che rimanga al passo con le aspettative degli utenti, in quanto la raccolta di opinioni e suggerimenti può aiutare gli operatori a sviluppare nuove funzionalità e migliorare quelle esistenti.

### 1.1.7 Admin

Dal punto di vista amministrativo, il sistema fornirà una serie di strumenti per monitorare lo stato della flotta di biciclette, gestire le segnalazioni di guasti e ottimizzare la distribuzione delle biciclette tra le varie stazioni.
Questi strumenti consentiranno agli admin di avere una visione chiara e aggiornata della situazione, facilitando interventi tempestivi e miglioramenti operativi.

### 1.1.8 Analytics

Altro aspetto fondamentale del sistema è la raccolta e l'analisi dei dati sull'utilizzo del servizio, che aiuta a pianificare eventuali miglioramenti del sistema.
L'analisi dei dati può rivelare tendenze nell'uso delle biciclette, permettendo agli amministratori di ottimizzare la distribuzione e la manutenzione delle biciclette.

### 1.1.9 Elenco richiesta

La richiesta si suddivide in 7 punti:

1. Permettere all’utente di effettuare la propria registrazione, in modo da rendere i dati persistenti per permettere all’utente di accedere in un secondo momento.
2. Mostrare una mappa con le varie stazioni di bike sharing, le loro disponibilità e i relativi percorsi ciclabili consigliati.
3. Permettere all’utente di gestire il proprio abbonamento e accedere al servizio di bike sharing, noleggiando le bici.
4. Permettere all’utente di inviare segnalazioni e feedback in merito al servizio.
5. Permettere agli admin di monitorare le varie stazioni e le relative biciclette.
6. Permettere agli admin di monitorare le biciclette, la loro disponibilità e le segnalazioni.
7. Permettere agli admin di raccogliere dati a fini statistici e funzionali dagli utenti.

## 1.2 Requisiti funzionali

| Codice | Nome | Descrizione |
|---|---|---|
| R00 | main | Avvia il programma, indirizza l'utente o l'admin al menu appropriato. |
| R01 | caricamento | Mostra una barra di caricamento che avanza. |
| R02 | printlogo | Mostra a schermo un asciiart di EcoRide. |
| R03 | printwelcome | Mostra a schermo il benvenuto in asciiart. |
| R04 | login | Permette all'utente di inserire le proprie credenziali. |
| R05 | verifica | Verifica la presenza delle credenziali nel file. |
| R06 | searchutente | Recupera un determinato dato da un record.<br>Simula il comportamento di una query SQL<br>"SELECT x<br>FROM utente<br>WHERE cf==codice_fiscale"<br>dove cf è il parametro di confronto univoco per la ricerca del record interessato e x è il campo che si vuole ottenere. |
| R07 | reg | Permette all'utente di inserire i propri dati in fase di registrazione. |
| R08 | verifica_email | Gestisce il passaggio di dati per la verifica dell'email. |
| R09 | create_complex_json | Attraverso la libreria cjson, vengono creati i vari oggetti json per le impostazioni dei parametri per gestire l'invio dell'email. |
| R10 | readFileContent | Legge il contenuto del file html e lo converte in stringa. |
| R11 | inviaEmail | Imposta tramite la libreria curl, i parametri per la richiesta https all'API di brevo. |
| R12 | randoma | Crea un codice randomico di 5 cifre. |
| R13 | paginaabbonamento | Presenta un sottomenu con le varie operazioni. |
| R14 | visabbonamento | Cerca l'abbonamento inerente al cf passato e stampa i dettagli dell'abbonamento. |

| | | |
|---|---|---|
| R15 | searchs | Trova il saldo dell'utente simulando il comportamento di una query SQL<br>"SELECT saldo<br>FROM utente<br>WHERE cf==codice_fiscale"<br>dove cf è il parametro di confronto univoco per la ricerca del record interessato e saldo è il campo che si vuole ottenere. |
| R16 | rinnovo | Fa scegliere all'utente tra abbonamento mensile e quello settimanale, calcola la scadenza e salva i dati sul file. |
| R17 | modificasaldo | Trova il saldo dell'utente, ne calcola la differenza col parametro e aggiorna il saldo. |
| R18 | cov | Calcola la differenza temporale tra quella reale e una impostata. |
| R19 | verificaabb | Verifica la validità dell'abbonamento di un utente. |
| R20 | paginacarta | Fa scegliere all'utente se aggiungere o modificare la carta. |
| R21 | aggiungicarta | Raccoglie i dati di pagamento dell'utente verificando che non siano già presenti nel file e che rispettino lo standard di lunghezza. |
| R22 | inscard | Stampa su una nuova riga del file carte.csv, i nuovi dati dell'utente. |
| R23 | modifica_carta | Prende i nuovi dati dell'utente, ne verifica lo standard di lunghezza e la presenza nel file, infine li inserisce in una nuova riga del file. |
| R24 | aperturamappa | Attraverso un comando cmd apre la pagina web di EcoRide contenente la mappa con le stazioni e i percorsi ciclabili. |
| R25 | pagina_bici | Indirizza gli utenti, in base alle verifiche, alle funzioni di restituzione o di noleggio. |
| R26 | verificatrans | Verifica che l'utente non abbia transazioni aperte. |
| R27 | noleggio | Prende in input l'id della stazione da cui l'utente vuole noleggiare una bici, con le varie verifiche. |
| R28 | verifica_disp_stazioni | Verifica che la disponibilità della stazione sia di almeno 1 bici. |

| | | |
|---|---|---|
| R29 | opbici | Ricerca una bici in una determinata stazione, effettuando le varie verifiche di disponibilità della bici e che non sia guasta. |
| R30 | opentransazione | Calcola data e ora di noleggio e di scadenza, salva su file col campo utilizzo a TRUE e invia all'utente le informazioni riguardanti il noleggio. |
| R31 | decrementa_stazione | Decrementa di uno la disponibilità in una determinata stazione, nel giusto formato. |
| R32 | stampastazioni | Fornisce a schermo una tabella dettagliata delle stazioni |
| R33 | calcolastazione | Calcola il numero delle stazioni disponibili. |
| R34 | restituzione | Permette all'utente di scegliere se restituire la bici o inviare una segnalazione. |
| R35 | segnalazione | Prende in input la descrizione fornita dall'utente e salva i dati nel file "segnalazioni.csv". |
| R36 | incrementabici | Rende la bici disponibile impostando il campo disponibilità a 0. |
| R37 | closetransazione | Imposta il campo "utilizzo" di "transazione.csv" a 0, e scrive l'id della stazione dove l'utente di quella transazione ha depositato la bici. |
| R38 | incrementastazioni | Incrementa la disponibilità delle bici in una determinata stazione di 1. |
| R39 | tokenumr | Trova all'interno di una stringa, il campo richiesto utilizzando come separatore la virgola. |
| R40 | printrecord | Separa la struct e restituisce il campo dell'utente richiesto. |
| R41 | searchcarta | Trova un determinato campo di una carta partendo da un dato univoco, salva tutti i dati della carta in una struct. |
| R42 | printcarta | Separa la struct e restituisce il dato richiesto. |
| R43 | ecosearchtra | Trova un determinato campo di una transazione trovata a partire da un dato univoco specificato e dal vincolo di utilizzo. |
| R44 | ecoFindIDSt | Cerca a partire dall'id della transazione, l'id della stazione di noleggio. |
| R45 | gestioneadmin | Permette di visualizzare un menu con le varie operazioni admin disponibili. |

| | | |
|---|---|---|
| R46 | impostaadmin | Cerca l'utente a partire dal suo codice fiscale, e imposta il campo admin a 1. |
| R47 | rimuoviadmin | Cerca l'utente tramite il suo codice fiscale e imposta il campo admin a 0. |
| R48 | visualizzadmin | Cerca tutti gli utenti aventi il campo admin uguale a 1 e li stampa a schermo. |
| R49 | gestionebici | Permette di visualizzare un menu con le varie operazioni disponibili. |
| R50 | inserisciBici | Calcola l'id della nuova bici, chiede all'admin l'id della stazione di deposito e inserisce alla fine del file una nuova riga dove stampa i vari parametri della bici. |
| R51 | visualizzaBici | Stampa a schermo tutte le bici che trova all'interno del file. |
| R52 | visualizzaBiciUso | Stampa a schermo tutte le bici che trova col vincolo di utilizzo. |
| R53 | gestione feedback | Stampa una tabella ordinata con tutti i feedback. |
| R54 | gestionesegnalazioni | Permette di visualizzare un menu contenente le varie operazioni disponibili. |
| R55 | disabilita | Rende una bici non disponibile per gli utenti, impostando il campo guasto a 1. |
| R56 | abilita | Rende una bici disponibile per gli utenti, impostando il campo guasto a 0. |
| R57 | ins | Inserisce i dati della stazione nel file. |
| R58 | gestionestazioni | Permette all'admin di inserire i dati di una nuova stazione. |
| R59 | gestionetrans | Permette di visualizzare le transazioni in modo ordinato per ogni stazione, calcolando il totale delle transazioni effettuate da ogni stazione. |
| R60 | printadmin | Permette di visualizzare un ascii art admin menu. |

**Tabella 1.2.1** Requisiti funzionali

## 1.3 Casi d’uso

| **Codice** | **Nome** | **Descrizione** |
|---|---|---|
| R00 | main | Avvia il programma, indirizza l’utente o l’admin al menu appropriato. |
| **Pre-condizioni** | Avvio del programma. | |
| **Post-condizioni** | **Avvio** | Il programma si avvia correttamente e continua il flusso di esecuzione. |
| | **Termine** | Il programma si chiude. |
| **Scenario di base** | L’utente avvia il programma. | |
| **Scenario alternativo** | L’utente decide di continuare con il programma già avviato. | |

**Tabella 1.3.1** caso d’uso main

| **Codice** | **Nome** | **Descrizione** |
|---|---|---|
| R01 | caricamento | Mostra una barra di caricamento che avanza. |
| **Pre-condizioni** | Il programma deve essere avviato. | |
| **Post-condizioni** | **Successo** | Il programma, all’avvio, mostra una barra di caricamento che avanza. |
| | **Fallimento** | Il programma non mostra una barra di caricamento. |
| **Scenario di base** | L’utente ha appena avviato il programma. | |

**Tabella 1.3.2** caso d’uso caricamento

| Codice | Nome | Descrizione |
|---|---|---|
| R02 | printlogo | Mostra a schermo un asciiart di EcoRide. |
| **Pre-condizioni** | Il programma deve essere avviato e deve aver mostrato la barra di caricamento. | |
| **Post-condizioni** | **Successo** | Il programma mostra un asciiart di EcoRide. |
| | **Fallimento** | Il programma non mostra l'asciiart di EcoRide. |
| **Scenario di base** | L'utente ha avviato il programma ed è stata già visualizzata la barra di caricamento. | |

**Tabella 1.3.3** caso d'uso printlogo

| Codice | Nome | Descrizione |
|---|---|---|
| R03 | printwelcome | Mostra a schermo il benvenuto in asciiart. |
| **Pre-condizioni** | Il programma deve essere avviato e deve aver mostrato un asciiart di EcoRide. | |
| **Post-condizioni** | **Successo** | Il programma mostra un asciiart di benvenuto. |
| | **Fallimento** | Il programma non mostra un asciiart di benvenuto. |
| **Scenario di base** | L'utente ha avviato il programma ed è stato già mostrato un asciiart del logo. | |

**Tabella 1.3.4** caso d'uso printwelcome

| Codice | Nome | Descrizione |
|---|---|---|
| R04 / R05 | login / verifica | Permette all'utente di inserire le proprie credenziali. / Verifica la presenza delle credenziali acquisite da login nel file "utente.csv". |
| **Pre-condizioni** | L'utente ha a disposizione un dispositivo di input. / Le credenziali dell'utente che accede devono essere già presenti nel file "utente.csv". | |
| **Post-condizioni** | **Successo** | L'utente accede al menu principale del software. |
| | **Fallimento** | L'utente ha inserito credenziali non presenti nel file "utente.csv". |
| **Scenario di base** | L'utente inserisce le credenziali della propria area riservata. | |
| **Scenario alternativo** | L'utente è un admin, quindi accede al menu admin. | |

**Tabella 1.3.5** caso d'uso login / verifica

| Codice | Nome | Descrizione |
|---|---|---|
| R06 | searchutente | Simula il comportamento di una query SQL<br>"SELECT x<br>FROM utente<br>WHERE cf==codice_fiscale"<br>dove cf è il parametro di confronto univoco per la ricerca del record interessato e x è il campo che si vuole ottenere. |
| **Pre-condizioni** | Il parametro di confronto deve essere inserito / Il file "utente.csv" deve essere aperto. | |
| **Post-condizioni** | **Successo** | Viene estrapolato il campo ricercato. |
| | **Fallimento** | Restituisce NULL |
| **Scenario di base** | Viene impostato un parametro specificando il campo dove effettuare il confronto, il campo da estrapolare e il tipo di dato da estrapolare. | |
| **Scenario alternativo** | Viene impostato un parametro differente dal codice fiscale per la ricerca. Se il dato non è univoco, viola i vincoli d'integrità referenziale e l'esito positivo della funzione non è garantito. | |

**Tabella 1.3.6** caso d'uso searchutente

| Codice | Nome | Descrizione |
|---|---|---|
| R07 | reg | Permette all'utente di inserire i propri dati in fase di registrazione. |
| **Pre-condizioni** | Verificare, attraverso searchutente, che le credenziali inserite dall'utente non siano presenti all'interno del file "utente.csv", il quale deve essere aperto. | |
| **Post-condizioni** | **Successo** | L'utente effettua la registrazione. |
| | **Fallimento** | La registrazione non viene effettuata. |
| **Scenario di base** | L'utente si interfaccia per la prima volta al software e non ha credenziali già registrate. | |

**Tabella 1.3.7** caso d'uso reg

| Codice | Nome | Descrizione |
|---|---|---|
| R08 | verifica_email | Gestisce il passaggio di dati per la verifica dell'email. |
| **Pre-condizioni** | Deve essere stata inserito un indirizzo email. | |
| **Post-condizioni** | **Successo** | L'email viene inviata correttamente. |
| | **Fallimento** | L'email non viene inviata. |
| **Scenario di base** | L'utente è in fase di registrazione e sta verificando la propria email. | |
| **Scenario alternativo** | L'utente inserisce in fase di registrazione un indirizzo email inesistente. | |

**Tabella 1.3.8** caso d'uso verifica_email

| Codice | Nome | Descrizione |
|---|---|---|
| R09 | create_complex_json | Attraverso la libreria cjson, vengono creati i vari oggetti json per le impostazioni dei parametri per gestire l'invio dell'email. |
| **Pre-condizioni** | Il codice di verifica deve essere stato già creato. | |
| **Post-condizioni** | **Successo** | Viene creato il payLoad contenente gli oggetti json col codice di verifica. |
| | **Fallimento** | Viene creato il payLoad contenente gli oggetti json senza codice di verifica. |
| **Scenario di base** | Viene creato il payLoad contenente il codice di verifica e i relativi oggetti e viene chiamata inviaEmail con parametro il payLoad. | |
| **Scenario alternativo** | Viene creato un payLoad contenente solo gli oggetti senza il codice di verifica, viene chiamata inviaEmail con parametro il payLoad. | |

**Tabella 1.3.9** caso d'uso create_complex_json

| Codice | Nome | Descrizione |
|---|---|---|
| R10 | readFileContent | Legge il contenuto del file html e lo converte in stringa. |
| **Pre-condizioni** | Deve essere presente un file html da leggere. | |
| **Post-condizioni** | **Successo** | Il contenuto del file html viene convertito in stringa e passato al software. |
| | **Fallimento** | Il contenuto del file html non viene convertito in stringa. |
| **Scenario di base** | L'utente è in fase di registrazione. | |

**Tabella 1.3.10** caso d'uso readFileContent

| Codice | Nome | Descrizione |
|---|---|---|
| R11 | inviaEmail | Imposta tramite la libreria curl, i parametri per la richiesta https all'API di brevo. |
| **Pre-condizioni** | Deve essere presente un file html da leggere. | |
| **Post-condizioni** | **Successo** | La funzione imposta i parametri per la richiesta https all'API di brevo e invia l'email. |
| | **Fallimento** | Non viene inviata alcuna email. |
| **Scenario di base** | L'utente in fase di registrazione. | |

**Tabella 1.3.11** caso d'uso inviaEmail

| Codice | Nome | Descrizione |
|---|---|---|
| R12 | randoma | Crea un codice randomico di 5 cifre. |
| **Pre-condizioni** | È stato inserito un indirizzo email da parte dell'utente in fase di registrazione. | |
| **Post-condizioni** | **Successo** | Viene generato un codice randomico di 5 cifre. |
| | **Fallimento** | Non viene generato alcun codice. |
| **Scenario di base** | L'utente è in fase di registrazione. | |

**Tabella 1.3.12** caso d'uso randoma

| Codice | Nome | Descrizione |
|---|---|---|
| R13 | paginaabbonamento | Presenta un sottomenu con le varie operazioni. |
| **Pre-condizioni** | L'utente ha scelto di gestire il proprio abbonamento. | |
| **Post-condizioni** | **Successo** | Viene visualizzata la pagina di gestione dell'abbonamento. |
| | **Fallimento** | Non viene visualizzata la pagina di gestione dell'abbonamento. |
| **Scenario di base** | L'utente gestisce l'abbonamento. | |

**Tabella 1.3.13** caso d'uso paginabbonamento

| Codice | Nome | Descrizione |
|---|---|---|
| R14 | visabbonamento | Cerca l'abbonamento inerente al cf passato e stampa i dettagli dell'abbonamento. |
| **Pre-condizioni** | Deve ricevere in input un cf esistente nel file "abbonamento.csv". | |
| **Post-condizioni** | **Successo** | Stampa i dettagli dell'abbonamento associato al cf in input, dopo averlo trovato. |
| | **Fallimento** | Se non trova il cf associato all'abbonamento, stampa solamente il saldo relativo all'account. |
| **Scenario di base** | L'utente consulta i dettagli dell'abbonamento da lui effettuato. | |

**Tabella 1.3.14** caso d'uso visabbonamento

| Codice | Nome | Descrizione |
|---|---|---|
| R15 | searchs | Trova il saldo dell'utente simulando il comportamento di una query SQL<br>"SELECT saldo<br>FROM utente<br>WHERE cf==codice_fiscale"<br>dove cf è il parametro di confronto univoco per la ricerca del record interessato e x è il campo che si vuole ottenere. |
| **Pre-condizioni** | Il file "utente.csv" deve contenere info sul saldo dell'account. | |
| **Post-condizioni** | **Successo** | Viene trovato il saldo dell'account dell'utente. |
| | **Fallimento** | Non viene trovato il saldo dell'account dell'utente. |
| **Scenario di base** | L'utente ha richiesto di visualizzare informazioni in merito al suo account, nello specifico il suo saldo. | |

**Tabella 1.3.15** caso d'uso searchs

| Codice | Nome | Descrizione |
|---|---|---|
| R16 | rinnovo | Fa scegliere all'utente tra abbonamento mensile e quello settimanale, calcola la scadenza e salva i dati sul file. |
| **Pre-condizioni** | Il file su cui salva i dati deve essere aperto. | |
| **Post-condizioni** | **Successo** | L'utente ha scelto la tipologia di abbonamento, ha visualizzato la data di scadenza. Tali dati sono stati salvati su file. |
| | **Fallimento** | L'utente non ha potuto scegliere la tipologia di abbonamento. Nessun dato è stato salvato. |
| **Scenario di base** | L'utente sceglie la tipologia di abbonamento al servizio. | |
| **Scenario alternativo** | L'utente ha già un abbonamento in corso. | |

**Tabella 1.3.16** caso d'uso rinnovo

| Codice | Nome | Descrizione |
|---|---|---|
| R17 | modificasaldo | Trova il saldo dell'utente, ne calcola la differenza col parametro e aggiorna il saldo. |
| **Pre-condizioni** | Il saldo deve essere già determinato. | |
| **Post-condizioni** | **Successo** | Il saldo viene aggiornato. |
| | **Fallimento** | Il saldo non viene aggiornato. |
| **Scenario di base** | L'utente ha scelto un abbonamento e le spese per lo stesso vengono sottratte dal saldo. | |
| **Scenario alternativo** | L'utente ha già un abbonamento in corso. | |

**Tabella 1.3.17** caso d'uso modificasaldo

| Codice | Nome | Descrizione |
|---|---|---|
| R18 | cov | Calcola la differenza temporale tra quella reale e una impostata. |
| **Pre-condizioni** | Un abbonamento deve essere attivo. | |
| **Post-condizioni** | **Successo** | Viene calcolata la durata residua dell'abbonamento. |
| | **Fallimento** | Non viene calcolata la durata residua dell'abbonamento. |
| **Scenario di base** | L'utente prova a rinnovare l'abbonamento, ma è già abbonato. | |

**Tabella 1.3.18** caso d'uso cov

| Codice | Nome | Descrizione |
|---|---|---|
| R19 | verificaabb | Verifica la validità dell'abbonamento di un utente. |
| **Pre-condizioni** | L'utente si trova nel menu abbonamento. | |
| **Post-condizioni** | **Successo** | Viene verificata l'effettiva validità dell'abbonamento, che è attivo. |
| | **Fallimento** | Non viene verificata la validità dell'abbonamento. |
| **Scenario di base** | L'utente sta provando a rinnovare l'abbonamento. | |
| **Scenario alternativo** | L'utente sta provando ad abbonarsi. | |

**Tabella 1.3.19** caso d'uso verificaabb

| Codice | Nome | Descrizione |
|---|---|---|
| R20 | paginacarta | Fa scegliere all'utente se aggiungere o modificare la carta. |
| **Pre-condizioni** | L'utente si trova nel menu abbonamento. | |
| **Post-condizioni** | **Successo** | L'utente accede alla pagina di modifica o aggiunta carta. |
| | **Fallimento** | L'utente non accede alla pagina di modifica o aggiunta carta. |
| **Scenario di base** | L'utente vuole gestire il metodo di pagamento del suo account. | |

**Tabella 1.3.20** caso d'uso paginacarta

| Codice | Nome | Descrizione |
|---|---|---|
| R21 | aggiungicarta | Raccoglie i dati di pagamento dell'utente verificando che non siano già presenti nel file e che rispettino lo standard di lunghezza. |
| **Pre-condizioni** | Devono esistere i dati di pagamento. | |
| **Post-condizioni** | **Successo** | Ha verificato che non siano già presenti i dati che ha raccolto e che siano conformi allo standard di lunghezza. |
| | **Fallimento** | Non viene verificata l'eventuale presenza dei dati di pagamento, né la loro conformità agli standard di lunghezza. |
| **Scenario di base** | L'utente sta inserendo un nuovo metodo di pagamento. | |
| **Scenario alternativo** | L'utente sta inserendo un metodo di pagamento non valido. | |

**Tabella 1.3.21** caso d'uso aggiungicarta

| Codice | Nome | Descrizione |
|---|---|---|
| R22 | inscard | Stampa su una nuova riga del file carte.csv, i nuovi dati dell'utente. |
| **Pre-condizioni** | Devono essere stati raccolti i dati della nuova carta. | |
| **Post-condizioni** | **Successo** | Sul file "carte.csv" è stata creata una nuova riga con i nuovi dati dell'utente. |
| | **Fallimento** | Sul file "carte.csv" non è stata creata una nuova riga con i nuovi dati dell'utente. |
| **Scenario di base** | L'utente ha inserito un nuovo metodo di pagamento. | |
| **Scenario alternativo** | L'utente ha inserito un metodo di pagamento non valido. | |

**Tabella 1.3.22** caso d'uso inscard

| Codice | Nome | Descrizione |
|---|---|---|
| R23 | modifica_carta | Prende i nuovi dati dell'utente, ne verifica lo standard di lunghezza e la presenza nel file, infine li inserisce in una nuova riga del file. |
| **Pre-condizioni** | Deve essere presente una riga da modificare. | |
| **Post-condizioni** | **Successo** | La riga individuata viene correttamente modificata |
| | **Fallimento** | Non viene individuata, né modificata una riga del file. |
| **Scenario di base** | L'utente sta modificando i dati del suo metodo di pagamento. | |
| **Scenario alternativo** | L'utente ha inserito nuovi dati di pagamento già presenti nel file o che non rispettano gli standard di lunghezza. | |

**Tabella 1.3.23** caso d'uso modifica_carta

| Codice | Nome | Descrizione |
|---|---|---|
| R24 | aperturamappa | Attraverso un comando cmd apre la pagina web di EcoRide contenente la mappa con le stazioni e i percorsi ciclabili. |
| **Pre-condizioni** | Deve essere online una pagina web di EcoRide. | |
| **Post-condizioni** | **Successo** | La pagina web viene correttamente aperta tramite browser. |
| | **Fallimento** | Non viene aperta alcuna pagina web. |
| **Scenario di base** | L'utente effettua il login a EcoRide. | |
| **Scenario alternativo** | L'utente decide di aprire nuovamente la pagina delle mappe. | |

**Tabella 1.3.24** caso d'uso aperturamappa

| Codice | Nome | Descrizione |
|---|---|---|
| R25 | pagina_bici | Indirizza gli utenti, in base alle verifiche, alla funzione di restituzione o di noleggio. |
| **Pre-condizioni** | L'utente deve essere loggato. | |
| **Post-condizioni** | **Successo** | L'utente viene reindirizzato alla pagina di restituzione o noleggio. |
| | **Fallimento** | L'utente non visualizza la pagina di restituzione o noleggio. |
| **Scenario di base** | L'utente abbonato vuole avviare il noleggio di una bici e non ne ha già una, pertanto viene reindirizzato a noleggio. | |
| **Scenario alternativo 1** | L'utente abbonato vuole restituire la bici noleggio, pertanto viene reindirizzato a restituzione. | |
| **Scenario alternativo 2** | L'utente non abbonato vuole noleggiare la bici, pertanto viene reindirizzato al menu principale. | |
| **Scenario alternativo 3** | L'utente abbonato vuole avviare il noleggio di una bici, ma ne ha già una, pertanto viene reindirizzato a restituzione. | |

**Tabella 1.3.25** caso d'uso pbici

| Codice | Nome | Descrizione |
|---|---|---|
| R26 | verificatrans | Verifica che l'utente non abbia transazioni aperte. |
| **Pre-condizioni** | L'utente deve essere loggato. | |
| **Post-condizioni** | **Successo** | La verifica sulle eventuali transazioni aperte dell'utente giunge a termine. |
| | **Fallimento** | La verifica sulle eventuali transazioni aperte dell'utente non giunge a termine. |
| **Scenario di base** | L'utente desidera noleggiare o restituire una bici. | |

**Tabella 1.3.26** caso d'uso verificatrans

| Codice | Nome | Descrizione |
|---|---|---|
| R27 | noleggio | Prende in input l'id della stazione da cui l'utente vuole noleggiare una bici, con le varie verifiche. |
| **Pre-condizioni** | L'id inserito dall'utente e relativo alla stazione da lui scelta deve essere corrispondente a una stazione. Tale stazione deve avere la disponibilità di almeno una bici. | |
| **Post-condizioni** | **Successo** | L'id della stazione viene correttamente acquisito. Viene verificata la disponibilità della stazione corrispondente a tale id. |
| | **Fallimento** | L'id della stazione non viene acquisito e non viene verificata la disponibilità di alcuna stazione. |
| **Scenario di base** | L'utente sceglie una stazione da cui prelevare una bici. | |

**Tabella 1.3.27** caso d'uso noleggio

| Codice | Nome | Descrizione |
|---|---|---|
| R28 | verifica_disp_stazione | Verifica che la disponibilità della stazione sia di almeno 1 bici. |
| **Pre-condizioni** | Il codice stazione deve riferirsi a una stazione valida e deve essere passato in input. | |
| **Post-condizioni** | **Successo** | Viene verificata la disponibilità di almeno 1 bici per la stazione in input. |
| | **Fallimento** | Non viene verificata la disponibilità di almeno 1 bici per la stazione in input. |
| **Scenario di base** | L'utente ha già scelto la stazione da cui prelevare una bici. | |
| **Scenario alternativo** | Il codice inserito dall'utente non corrisponde ad alcuna stazione. | |

**Tabella 1.3.28** caso d'uso verdispst

| Codice | Nome | Descrizione |
|---|---|---|
| R29 | opbici | Ricerca una bici in una determinata stazione, effettuando le varie verifiche di disponibilità della bici e che non sia guasta. |
| **Pre-condizioni** | La stazione deve avere almeno una bici disponibile. | |
| **Post-condizioni** | **Successo** | Trova la bici disponibile in una determinata stazione, ne controlla eventuali guasti. Avvia il processo di registrazione della bici come "NOLEGGIATA". Imposta "NON DISPONIBILE" come stato della bici. |
| | **Fallimento** | Non trova bici disponibili e non imposta alcuno stato della bici. |
| **Scenario di base** | L'utente ha già scelto la stazione da cui prelevare una bici. La stazione ha l'effettiva disponibilità. | |
| **Scenario alternativo** | L'utente non ha confermato il noleggio della bici. | |

**Tabella 1.3.29** caso d'uso opbici

| Codice | Nome | Descrizione |
|---|---|---|
| R30 | opentransazione | Calcola data e ora di noleggio e di scadenza, salva su file col campo "utilizzo" a "TRUE" e invia all'utente le informazioni riguardanti il noleggio. |
| **Pre-condizioni** | L'utente deve aver confermato il noleggio. | |
| **Post-condizioni** | **Successo** | Viene calcolata la data e l'ora di noleggio e di scadenza, viene salvato su file lo stato della transazione a "TRUE". Invia all'utente le informazioni riguardanti il noleggio. |
| | **Fallimento** | Non viene calcolata alcuna scadenza, né viene salvato su file lo stato della transazione. |
| **Scenario di base** | L'utente è in fase di noleggio della bici. | |

**Tabella 1.3.30** caso d'uso opentransazione

| Codice | Nome | Descrizione |
|---|---|---|
| R31 | decrementa_stazione | Decrementa di uno la disponibilità in una determinata stazione, nel giusto formato. |
| **Pre-condizioni** | Deve essere stata noleggiata una bici da un utente. | |
| **Post-condizioni** | **Successo** | La disponibilità della stazione viene decrementata di 1. |
| | **Fallimento** | La disponibilità della stazione rimane la medesima. |
| **Scenario di base** | L'utente ha scelto la stazione e ha ricevuto il messaggio di conferma di noleggio della bici. | |

**Tabella 1.3.31** caso d'uso decrementa_stazione

| Codice | Nome | Descrizione |
|---|---|---|
| R32 | stampastazioni | Fornisce a schermo una tabella dettagliata delle stazioni. |
| **Pre-condizioni** | L'utente è abilitato al noleggio. | |
| **Post-condizioni** | **Successo** | Le stazioni vengono correttamente e dettagliatamente stampate. |
| | **Fallimento** | Non viene stampata la tabella delle stazioni. |
| **Scenario di base** | L'utente intende noleggiare una bici. | |

**Tabella 1.3.32** caso d'uso stampastazioni

| Codice | Nome | Descrizione |
|---|---|---|
| R33 | calcolastazione | Calcola il numero delle stazioni disponibili. |
| **Pre-condizioni** | Il file “stazioni.csv” deve essere presente nella directory. | |
| **Post-condizioni** | **Successo** | Viene calcolato il numero delle stazioni disponibili. |
| | **Fallimento** | Non viene calcolato il numero di stazioni disponibili. |
| **Scenario di base** | L’utente ha avviato il noleggio. | |
| **Scenario alternativo** | L’admin ha introdotto una nuova stazione. | |

**Tabella 1.3.33** caso d’uso calcolastazione

| Codice | Nome | Descrizione |
|---|---|---|
| R34 | restituzione | Permette all’utente di scegliere se restituire la bici o inviare una segnalazione. |
| **Pre-condizioni** | Il noleggio è stato già avviato. | |
| **Post-condizioni** | **Successo** | Viene chiesto all’utente se restituire la bici o inviare una segnalazione. |
| | **Fallimento** | Il menu per permettere all’utente di scegliere tra la restituzione e la segnalazione non viene visualizzato. |
| **Scenario di base** | L’utente è in fase di termine del noleggio. | |
| **Scenario alternativo** | L’utente vuole effettuare una segnalazione. | |

**Tabella 1.3.34** caso d’uso restituzione

| Codice | Nome | Descrizione |
|---|---|---|
| R35 | segnalazione | Prende in input la descrizione fornita dall'utente e salva i dati nel file "segnalazioni.csv". |
| **Pre-condizioni** | Il noleggio è in corso, è stato avviato il processo di restituzione. | |
| **Post-condizioni** | **Successo** | La descrizione della segnalazione viene salvata nel file "segnalazioni.csv". |
| | **Fallimento** | La descrizione non viene presa in input, né viene salvata nel file "segnalazioni.csv". |
| **Scenario di base** | L'utente desidera trasmettere una segnalazione. | |

**Tabella 1.3.35** caso d'uso segnalazione

| Codice | Nome | Descrizione |
|---|---|---|
| R36 | incrementa_bici | Rende la bici disponibile impostando il campo disponibilità a 0. |
| **Pre-condizioni** | Deve essere terminato il noleggio relativo a quella bici. | |
| **Post-condizioni** | **Successo** | La disponibilità della bici viene modificata nel file "bici.csv". |
| | **Fallimento** | La bici rimane non disponibile. |
| **Scenario di base** | L'utente ha terminato il noleggio. | |

**Tabella 1.3.36** caso d'uso incrementa_bici

| Codice | Nome | Descrizione |
|---|---|---|
| R37 | closetransazione | Imposta il campo “utilizzo” di “transazione.csv” a 0, e scrive l’id della stazione dove l’utente di quella transazione ha depositato la bici. |
| **Pre-condizioni** | La bici deve essere stata depositata. | |
| **Post-condizioni** | **Successo** | Il campo “utilizzo” di “transazione.csv” relativo a quella bici, viene impostato a 0. Nel file viene inserito l’id della stazione a cui tale bici è stata riconsegnata. |
| | **Fallimento** | Il campo “utilizzo” di “transazione.csv” non viene impostato a 0. Nel file non viene inserito il corretto id della stazione a cui tale bici è stata riconsegnata. |
| **Scenario di base** | L’utente, dopo il termine del noleggio, attende la chiusura della transazione. | |

**Tabella 1.3.37** caso d’uso closetransazione

| Codice | Nome | Descrizione |
|---|---|---|
| R38 | incrementa_stazione | Incrementa la disponibilità delle bici in una determinata stazione di 1. |
| **Pre-condizioni** | Deve essere stata depositata a termine del noleggio una bici in una stazione EcoRide. | |
| **Post-condizioni** | **Successo** | La disponibilità della stazione alla quale è stata riconsegnata la bici viene incrementata di 1. |
| | **Fallimento** | Il numero di bici disponibili per quella stazione rimane il medesimo. |
| **Scenario di base** | La stazione ha ricevuto una bici restituita da un utente. | |
| **Scenario alternativo** | L’admin ha aggiunto una bici in una stazione. | |

**Tabella 1.3.38** caso d’uso incrementa_stazione

| Codice | Nome | Descrizione |
|---|---|---|
| R39 | tokenumr | Trova all'interno di una stringa, il campo richiesto utilizzando come separatore la virgola. |
| **Pre-condizioni** | Deve essere presente una stringa in input, con valori separati da virgole. Il numero del campo non può essere maggiore al numero dei campi presenti nella stringa. | |
| **Post-condizioni** | **Successo** | Viene estrapolato il campo ricercato. |
| | **Fallimento** | Restituisce NULL. |
| **Scenario di base** | La funzione itera n volte, fino a trovare il token in posizione n. | |
| **Scenario alternativo** | Il numero di iterazioni è maggiore del numero di campi e la funzione restituisce NULL. | |

**Tabella 1.3.39** caso d'uso tokenumr

| Codice | Nome | Descrizione |
|---|---|---|
| R40 | printrecord | Separa la struct e restituisce il campo dell'utente richiesto. |
| **Pre-condizioni** | L'utente richiesto esiste. | |
| **Post-condizioni** | **Successo** | In base al campo richiesto, stampa il contenuto relativo all'utente. |
| | **Fallimento** | Non stampa a schermo il campo richiesto. |
| **Scenario di base** | Searchutente chiama la funzione printrecord per stampare il campo trovato. | |

**Tabella 1.3.40** caso d'uso printrecord

| Codice | Nome | Descrizione |
|---|---|---|
| R41 | searchcarta | Trova un determinato campo di una carta partendo da un dato univoco, salva tutti i dati della carta in una struct. |
| **Pre-condizioni** | Deve essere presente una stringa in input, contenente un dato univoco, associato ad una carta. | |
| **Post-condizioni** | **Successo** | Vengono estrapolati i dati della carta associata al dato in input e successivamente salvati in una struct. |
| | **Fallimento** | Se non trova nessuna carta, restituisce “NON CARTA”. |
| **Scenario di base** | L’utente inserisce un campo univoco, la funzione salva i dati della carta associata a quel campo in una struct. | |
| **Scenario alternativo** | L’utente inserisce come campo univoco il numero della carta, anziché il cf. | |

**Tabella 1.3.41** caso d’uso searchcarta

| Codice | Nome | Descrizione |
|---|---|---|
| R42 | printcarta | Separa la struct e restituisce il dato richiesto. |
| **Pre-condizioni** | La carta richiesta esiste. | |
| **Post-condizioni** | **Successo** | In base al campo richiesto, stampa il contenuto relativo alla carta. |
| | **Fallimento** | Non stampa a schermo il campo richiesto. |
| **Scenario di base** | Searchcarta chiama la funzione printcarta per stampare il campo trovato. | |

**Tabella 1.3.42** caso d’uso printcarta

| Codice | Nome | Descrizione |
|---|---|---|
| R43 | ecosearchtra | Trova un determinato campo di una transazione trovata a partire da un dato univoco specificato e dal vincolo di utilizzo. |
| **Pre-condizioni** | Il file “transazione.csv” deve essere in directory. | |
| **Post-condizioni** | **Successo** | Viene trovato un determinato campo di una transazione in base al dato univoco e dal vincolo di utilizzo specificati. |
| | **Fallimento** | Non viene trovato alcun campo. |
| **Scenario di base** | Si vuole trovare un dato di una transazione attraverso l’id. | |
| **Scenario alternativo** | Si vuole trovare un dato di una transazione attraverso il cf. | |

**Tabella 1.3.43** caso d’uso ecosearchtra

| Codice | Nome | Descrizione |
|---|---|---|
| R44 | ecoFindIDSt | Cerca a partire dall’id della transazione, l'id della stazione di noleggio. |
| **Pre-condizioni** | Il file “transazione.csv” deve essere in directory. | |
| **Post-condizioni** | **Successo** | Viene trovato l’id di una stazione a partire dall’id della transazione. |
| | **Fallimento** | Non viene trovato alcun id. |
| **Scenario di base** | Si vuole trovare un dato di una transazione attraverso l’id. | |

**Tabella 1.3.44** caso d’uso ecoFindIDSt

| Codice | Nome | Descrizione |
|---|---|---|
| R45 | gestioneadmin | Permette di visualizzare un menu con le varie operazioni admin disponibili. |
| **Pre-condizioni** | L'utente deve essere un admin. | |
| **Post-condizioni** | **Successo** | Stampa a video il menu con le varie operazioni admin. |
| | **Fallimento** | Non viene stampato il menu admin. |
| **Scenario di base** | L'admin ha effettuato l'accesso. | |

**Tabella 1.3.45** caso d'uso gestioneadmin

| Codice | Nome | Descrizione |
|---|---|---|
| R46 | impostaadmin | Cerca l'utente a partire dal suo codice fiscale, e imposta il campo admin a 1. |
| **Pre-condizioni** | L'utente deve essere presente nel file "utente.csv". | |
| **Post-condizioni** | **Successo** | Il campo admin dell'utente viene impostato a 1 |
| | **Fallimento** | Il campo admin dell'utente rimane invariato |
| **Scenario di base** | Si vuole impostare un nuovo admin | |

**Tabella 1.3.46** caso d'uso impostaadmin

| Codice | Nome | Descrizione |
|---|---|---|
| R47 | rimuoviadmin | Cerca l'utente tramite il suo codice fiscale e imposta il campo admin a 0. |
| **Pre-condizioni** | L'utente da trovare deve essere admin. | |
| **Post-condizioni** | **Successo** | L'utente admin viene trovato e il suo campo "admin" viene portato a 0. |
| | **Fallimento** | L'admin non viene trovato. |
| **Scenario di base** | Si vuole cambiare la tipologia di account a un admin. | |

**Tabella 1.3.47** caso d'uso rimuoviadmin

| Codice | Nome | Descrizione |
|---|---|---|
| R48 | visualizzadmin | Cerca tutti gli utenti aventi il campo admin uguale a 1 e li stampa a schermo. |
| **Pre-condizioni** | Il file "utente.csv" deve essere in directory. | |
| **Post-condizioni** | **Successo** | Vengono stampati tutti gli utenti aventi il campo admin impostato a 1 |
| | **Fallimento** | Non viene stampato nulla |
| **Scenario di base** | L'admin vuole controllare la lista degli amministratori. | |

**Tabella 1.3.48** caso d'uso visualizzadmin

| Codice | Nome | Descrizione |
|---|---|---|
| R49 | gestionebici | Permette di visualizzare un menu con le varie operazioni disponibili. |
| **Pre-condizioni** | L'utente deve essere un admin. | |
| **Post-condizioni** | **Successo** | Viene visualizzato il menu con le varie operazioni sulle bici. |
| | **Fallimento** | Non viene stampato il menu di gestione delle bici. |
| **Scenario di base** | L'admin vuole accedere al menu per gestire le bici. | |

**Tabella 1.3.49** caso d'uso gestionebici

| Codice | Nome | Descrizione |
|---|---|---|
| R50 | inserisciBici | Calcola l'id della nuova bici, chiede all'admin l'id della stazione di deposito e inserisce alla fine del file una nuova riga dove stampa i vari parametri della bici. |
| **Pre-condizioni** | Il file "bici.csv" deve essere in directory. | |
| **Post-condizioni** | **Successo** | Viene inserita una nuova bici in una determinata stazione. |
| | **Fallimento** | La bici non viene inserita. |
| **Scenario di base** | L'admin vuole inserire una bici alla lista. | |

**Tabella 1.3.50** caso d'uso inserisciBici

| Codice | Nome | Descrizione |
|---|---|---|
| R51 | visualizzaBici | Stampa a schermo tutte le bici che trova all'interno del file. |
| **Pre-condizioni** | Il file "bici.csv" deve essere in directory. | |
| **Post-condizioni** | **Successo** | Viene stampato a schermo l'elenco delle bici presenti nel file "bici.csv". |
| | **Fallimento** | Non viene stampata alcuna bici a schermo. |
| **Scenario di base** | Si vuole ottenere a schermo un elenco di tutte le bici esistenti nel sistema EcoRide. | |

**Tabella 1.3.51** caso d'uso visualizzaBici

| Codice | Nome | Descrizione |
|---|---|---|
| R52 | visualizzaBiciUso | Stampa a schermo tutte le bici che trova col vincolo di utilizzo. |
| **Pre-condizioni** | Il file “bici.csv” deve essere in directory. | |
| **Post-condizioni** | **Successo** | Vengono stampate tutte le bici in uso. |
| | **Fallimento** | Non viene stampato nulla. |
| **Scenario di base** | Si vuole controllare la lista delle bici in utilizzo. | |

**Tabella 1.3.52** caso d’uso visualizzaBiciUso

| Codice | Nome | Descrizione |
|---|---|---|
| R53 | gestionefeedback | Stampa una tabella ordinata con tutti i feedback. |
| **Pre-condizioni** | Il file “feedbacks.csv” deve essere in directory. | |
| **Post-condizioni** | **Successo** | I feedback vengono stampati ordinatamente in una tabella. |
| | **Fallimento** | Non viene stampato alcun feedback. |
| **Scenario di base** | Si vuole ottenere un elenco ordinato dei feedback. | |

**Tabella 1.3.53** caso d’uso gestionefeedback

| Codice | Nome | Descrizione |
|---|---|---|
| R54 | gestionesegnalazioni | Permette di visualizzare un menu contenente le varie operazioni disponibili. |
| **Pre-condizioni** | L’utente deve essere un admin | |
| **Post-condizioni** | **Successo** | Viene stampato un menu contenente le varie operazioni |
| | **Fallimento** | Non viene stampato nulla e l’admin viene reindirizzato al menu admin. |
| **Scenario di base** | L’admin vuole accedere al menu segnalazioni. | |

**Tabella 1.3.54** caso d’uso gestionesegnalazioni

| Codice | Nome | Descrizione |
|---|---|---|
| R55 | disabilita | Rende una bici non disponibile per gli utenti, impostando il campo guasto a 1. |
| **Pre-condizioni** | L'utente deve essere admin. | |
| **Post-condizioni** | **Successo** | La bici viene impostata come guasta, modificando il campo guasto a 1. |
| | **Fallimento** | La bici guasta rimarrà impostata come funzionante. |
| **Scenario di base** | L'admin, a seguito di segnalazioni, decide di classificare come guasta una bici, per renderla inutilizzabile. | |

**Tabella 1.3.55** caso d'uso disabilita

| Codice | Nome | Descrizione |
|---|---|---|
| R56 | abilita | Rende una bici disponibile per gli utenti, impostando il campo guasto a 0. |
| **Pre-condizioni** | L'utente deve essere un admin | |
| **Post-condizioni** | **Successo** | La bici impostata come guasta, viene reimpostata a funzionante modificando il campo guasto a 0. |
| | **Fallimento** | Non vengono apportate modifiche alla bici impostata come guasta. |
| **Scenario di base** | L'admin, a seguito di riparazioni della bici guasta, la reimposta funzionante. | |

**Tabella 1.3.56** caso d'uso abilita

| Codice | Nome | Descrizione |
|---|---|---|
| R57 | ins | Inserisce i dati della stazione nel file. |
| **Pre-condizioni** | L'utente deve essere admin. | |
| **Post-condizioni** | **Successo** | Il file "stazioni.csv" viene popolato. |
| | **Fallimento** | Non saranno inseriti dati nel file "stazioni.csv". |
| **Scenario di base** | L'admin desidera inserire i dati relativi alle stazioni nella tabella "stazioni.csv". | |

**Tabella 1.3.57** caso d'uso ins

| Codice | Nome | Descrizione |
|---|---|---|
| R58 | gestionestazioni | Permette all'admin di inserire i dati di una nuova stazione. |
| **Pre-condizioni** | L'utente deve essere un admin. | |
| **Post-condizioni** | **Successo** | Saranno inseriti in "stazioni.csv" i dati relativi a una nuova stazione. |
| | **Fallimento** | Non sarà inserito alcun dato relativo alla nuova stazione. |
| **Scenario di base** | L'admin desidera aggiungere una nuova stazione alla tabella "stazioni.csv". | |

**Tabella 1.3.58** caso d'uso gestionestazioni

| Codice | Nome | Descrizione |
|---|---|---|
| R59 | gestionetrans | Permette di visualizzare le transazioni in modo ordinato per ogni stazione, calcolando il totale delle transazioni effettuate da ogni stazione. |
| **Pre-condizioni** | L'utente deve essere un admin | |
| **Post-condizioni** | **Successo** | Viene stampata una lista con tutte le transazioni ordinate in base al codice stazione. |
| | **Fallimento** | Non vengono stampate le transazioni |
| **Scenario di base** | L'admin, vuole consultare l'andamento dei noleggi per ogni stazione. | |

**Tabella 1.3.59** caso d'uso gestionetrans

| Codice | Nome | Descrizione |
|---|---|---|
| R60 | printadmin | Permette di visualizzare un ascii art admin menu. |
| **Pre-condizioni** | L'utente deve essere un admin. | |
| **Post-condizioni** | **Successo** | Viene stampato un ascii art admin menu. |
| | **Fallimento** | Non viene stampato a schermo l'ascii art. |
| **Scenario di base** | L'admin ha effettuato l'accesso in piattaforma. | |

**Tabella 1.3.60** caso d'uso printadmin

# 2. Progettazione

Il programma al suo avvio, carica le librerie principali e stampa a schermo una barra di caricamento. Successivamente vengono stampati gli ascii art del logo e del benvenuto, ponendo l'utente dinanzi alla scelta tra la registrazione e il login con un account esistente.

La fase di registrazione comprende l'acquisizione dei dati personali dell'utente il quale, attraverso la registrazione, accetta i termini e le condizioni d'uso, nonché il trattamento dei dati personali. Dopodichè verrà generato un codice casuale che, tramite l'API di Brevo configurata all'interno del programma attraverso le librerie esterne cJson e curl, verrà inviato alla email inserita dall'utente, per scongiurare l'utilizzo del sistema da parte di bot, verificando che l'indirizzo email inserito sia corretto ed esistente.
Una volta ultimata la registrazione, l'utente dovrà effettuare il login per accedere al menu principale.

Nel menu principale viene stampato il logo del programma e l'ascii art di "MENU".
L'utente potrà scegliere se noleggiare una bici, gestire il proprio abbonamento, gestire il proprio metodo di pagamento o aprire la mappa.

1. Pagina noleggio-restituzione bici: attraverso le verifiche dell'abbonamento e della transazione, l'utente verrà reindirizzato automaticamente alle seguenti pagine:

    a. Pagina di noleggio.

        Se non ha transazioni aperte e ha un abbonamento valido, verrà mostrata all'utente una tabella contenente le varie stazioni con le relative disponibilità di bici.
        Inserito il codice della stazione, il programma fornirà le informazioni inerenti al numero della bici e al tempo disponibile e chiederà all'utente di confermare il noleggio.
        Infine, il programma mostrerà tutte le informazioni riguardanti la transazione.

    b. Pagina di deposito.

        Se ha una transazione aperta, viene chiesto all'utente se vuole depositare una bici o inviare una segnalazione; depositata la bici in una stazione, l'utente potrà inviare un feedback composto da un voto (1-5) e un commento.

    c. Se l'utente vorrà inviare una segnalazione, verrà reindirizzato a un'altra pagina dove potrà descrivere l'entità del problema.

d. Nel caso in cui l'utente non sia abbonato, il sistema glielo comunica e gli impedisce di accedere al servizio di noleggio.

2. Pagina abbonamento: in questa pagina l'utente potrà visualizzare lo stato e le informazioni riguardanti il suo abbonamento corrente, oppure scegliere l'opzione del rinnovo, valutando una soluzione mensile o settimanale. All'utente verrà addebitato il costo dell'abbonamento scelto dal saldo dell'account.

3. Pagina carta: l'utente potrà inserire una carta solo se non ne ha già inserita una, se non è presente nel file e se rispetta gli standard di lunghezza. In alternativa, potrà modificare la propria carta inserendo nuovi dati, i quali non devono essere già presenti nel file; questi ultimi dovranno rispettare gli standard di lunghezza.

4. Apertura mappa: attraverso il CMD viene inviato un comando che indirizzi l'utente al sito web di EcoRide contenente la mappa con le varie stazioni e i percorsi ciclabili consigliati.

5. Logout: viene chiesto all'utente se ha intenzione di effettuare il logout e, in caso affermativo, di chiudere il programma.

6. Pagina admin: qualora l'utente fosse impostato come admin, non appena effettuato l'accesso, gli verrà mostrato il menu admin con le seguenti operazioni disponibili:

    a. Gestione admin.

        In questa pagina verrà mostrato un sottomenu per mezzo del quale l'admin potrà aggiungere i permessi admin a un altro utente, rimuovere tali permessi ad un altro admin e visualizzare tutta la lista degli admin.

    b. Gestione bici.

        Qui verrà mostrato un sottomenu, tramite il quale l'admin potrà inserire una bici in una determinata stazione, visualizzare la lista completa delle bici presenti nel sistema o visualizzare la lista delle bici correntemente in uso.

    c. Gestione segnalazioni.

        L'admin potrà scegliere, attraverso un sottomenu, di visualizzare le varie segnalazioni, impostare una determinata bici come guasta, o reimpostare una bici guasta al suo stato normale.

    d. Inserisci stazioni.

Qui l’admin potrà inserire una nuova stazione con i relativi dati (disponibilità bici inizializzata a 0).

e. Visualizza transazioni.

   Mostra all’admin lo storico delle transazioni per stazione, indicando il numero totale di transazioni per stazione.

f. Visualizza Feedback.

   Mostra all’admin una tabella contenente tutti i feedback.

## 2.2 Principali variabili, strutture dati e file

Nella tabella seguente verranno indicate e descritte le principali variabili, struct e file utilizzati nel codice sorgente. Per la lista completa, è possibile consultare la documentazione doxygen presente nella repository.

NOTA: Come si può evincere dalla tabella seguente, molte variabili originariamente di tipo booleano, sono state dichiarate come interi. Questo per semplificare la restituzione dei dati delle varie funzioni.

| Nome | Tipologia | Descrizione | Tipi/Campi/Valori |
|---|---|---|---|
| utente | struct | Tipo di dato usato per raccogliere i dati dell'utente in fase di registrazione | Nome: char[30];<br>Cognome: char[30];<br>Email: char[30];<br>N_tel: char [10];<br>CF: char[15];<br>Password:char[30];<br>Admin: bool;<br>Saldo:float; |
| abbonamento | struct | Contiene le informazioni riguardanti l'abbonamento di un utente | cf: char[16]<br>tipo: char [10]<br>data:char[50]<br>scadenza:char[50]<br>carta: char[50] |
| carta | struct | Tipo di dato usato per raccogliere i dati della carta dell'utente in fase di inserimento | Numerocarta: char[16]<br>Mese: char [2]<br>Anno: char [2]<br>Cvv: char [3] |
| code | variabile locale | Raccoglie al suo interno il codice casuale da inviare per email all'utente | Intero |
| emailver | variabile locale | Contiene l'esito della funzione usata per verificare che le credenziali del nuovo utente non siano già presenti nel file | Int:0-1 |
| cfver | variabile locale | Contiene l'esito della funzione usata per verificare che il codice fiscale del nuovo utente non sia presente all'interno del file | Int:0-1 |
| email | variabile locale | Prende in input l'email dell'utente | char*[30] |
| password | variabile locale | Prende in input la password inserita dall'utente | char*[30] |
| res | variabile locale | Contiene l'esito della ricerca dei parametri email e password all'interno del file | Intero:0-1 |

| | | | |
|---|---|---|---|
| cf | variabile globale | Esito della ricerca del codice fiscale dell'utente a partire dall'email | char*[16] |
| buffer | variabile locale | Contiene la riga del file estratta da fgets | char[1024] |
| token | variabile locale | contiene il token estratto dalla stringa | char*[50] |
| email | variabile locale | Contiene l'email inserita dall'utente in fase di accesso | char*[30] |
| password | variabile locale | Contiene la password inserita dall'utente in fase d'accesso | char[30] |
| abbonamento.csv | FILE | File csv contenente tutti gli abbonamenti | "CF, tipo, data, scadenza, numero carta" |
| utente.csv | FILE | File csv contenente tutte le informazioni dell'utente | "email, password, nome, cognome, n_telefono, cf, admin, saldo" |
| carte.csv | FILE | File csv contenente tutte le informazioni delle carte | "Numero carta, scadenza_mese, scadenza_anno, cvv, CF" |
| bici.csv | FILE | File csv contenente tutte le informazioni delle bici | "ID_bici, guasto, disponibile, ID_stazione" |
| feedbacks.csv | FILE | File csv contenente tutti i feedback | "CF, voto, commento, data commento" |
| segnalazione.csv | FILE | File csv contenente tutte le segnalazioni | "ID_Segnalazione, CF, data noleggio, data segnalazione, segnalazione" |
| stazioni.csv | FILE | File csv contenente tutte le informazioni delle stazioni | "Nome, Indirizzo, disponibilità, codice" |
| transazione.csv | FILE | File csv contenente tutte le informazioni riguardanti le transazioni | "ID_transazione, CF, ID_bici, data noleggio, data restituzione, uso, ID_stazione_noleggio, ID_stazione_deposito" |

| | | | |
|---|---|---|---|
| t | time_t | Contiene data e ora correnti | hh:mm:ss gg//mm//yyy |
| scadenza | struct tm | Contiene una scadenza impostata | hh:mm:ss gg//mm//yyy |
| choose | variabile locale | Contiene la scelta dell'utente | intero |
| token | variabile locale | Contiene il token n di una stringa | char* |
| codice | variabile locale | Contiene l'id stazione del noleggio/deposito | intero |

## 2.3 Librerie e funzioni

Si propone di seguito un elenco completo delle librerie create durante lo sviluppo del software con le relative funzioni.

1. **standard c:**
    a. stdio.h
    b. stdlib.h
    c. string.h
    d. time.h
    e. stdbool.h
    f. unistd.h

2. **esterne:**
    a. cJson.h
    b. curl.h

3. **addc.h**
    a. void aggiungicarta(char*);
    b. void modifica(char*);

4. **admin.h**
    a. void gestioneadmin();
    b. void gestionebici();
    c. void gestioneabbonamenti();
    d. void gestionefeedback();
    e. void gestionesegnalazioni();
    f. void gestionestazioni();
    g. void gestionetrans();

5. **auth.h**
    a. int randoma();
    b. void reg();
    c. char* login();

6. **ecoascii.h**
    a. void printlogo();
    b. void printmenu();
    c. void printwelcome();
    d. void caricamento();
    e. void printadmin();

7. **ecoquery.h**
    a. char* searchutente(char*, int, int);
    b. char* searchcarta(char*, int, int);
    c. char* tokenumr(int n, char *buffer);
    d. char* ecosearchtra(char*parametro, int n,int ret,int uso);
    e. char* ecofindidSt(char*id_trans);
    f. int verificaabb(char*cf);
    g. void incrementa_stazione(int);
    h. void segnalazione(char*,char*);

8. **email.h**
    a. void verifica_email(char *, char *, int);

9. **mappa.h**
    a. void aperturamappa();

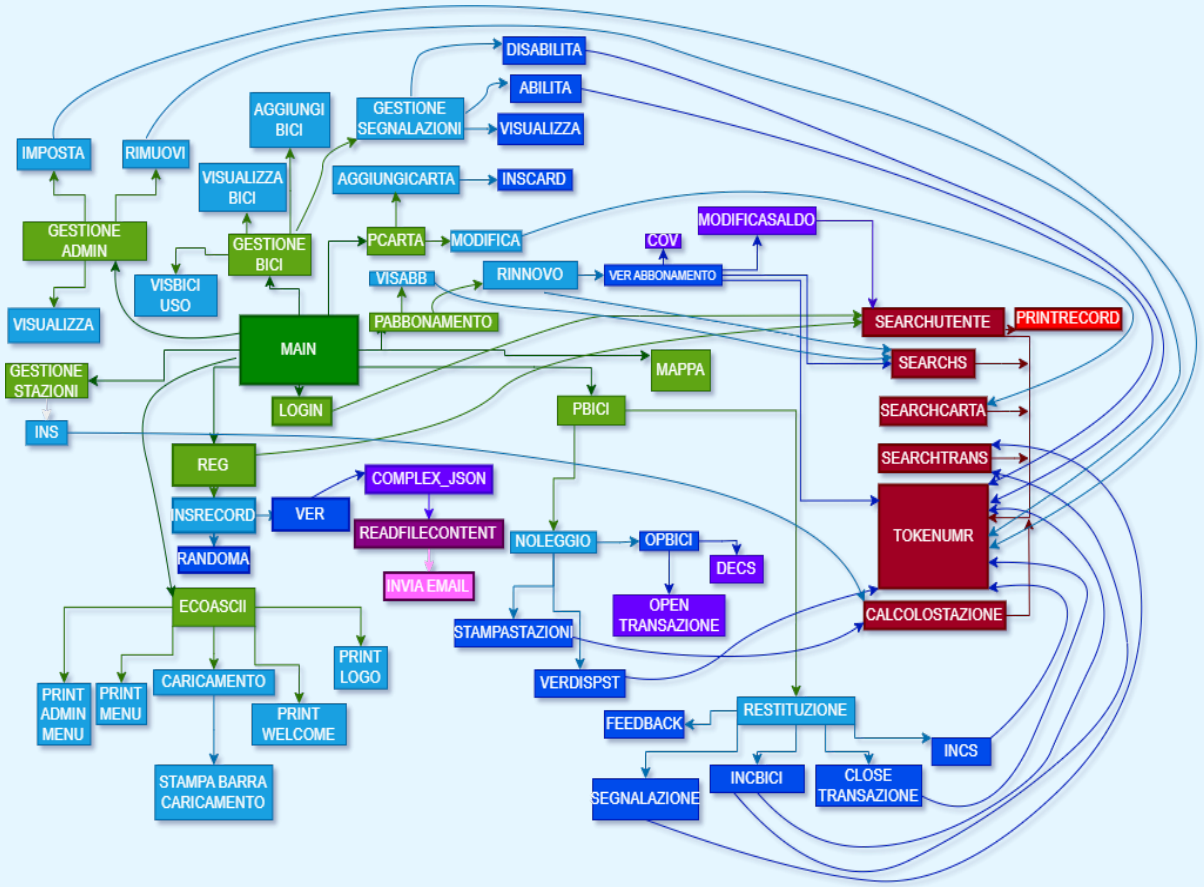
10. **menupages.h**
    a. void paginaabbonamento(char*);
    b. void paginapagamento(char*);
    c. void pagina_bici(char*);
    d. void paginacarta(char*);
    e. void pbici(char*);

11. **saldo.h**
    a. char* searchs(char*);
    b. void modificasaldo(char*,char*);
    c. void registraFeedback(char*);

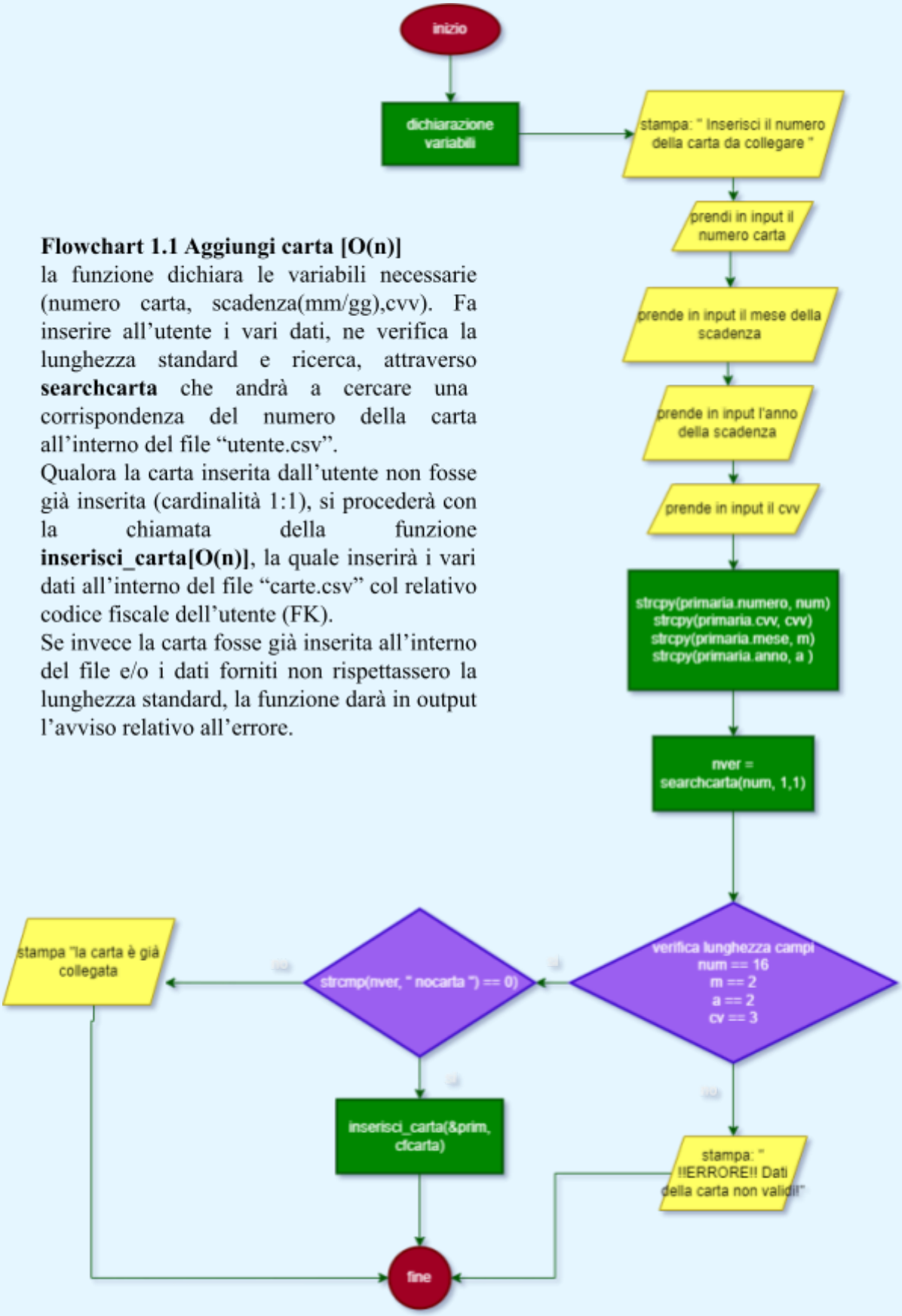
12. **utilityadmin.h**
    a. void impostaadmin();
    b. void visualizzadmin();
    c. void rimuoviadmin();
    d. void inserisciBici();
    e. void visualizzaBici();
    f. void visualizzaBiciUso();
    g. void disabilita();
    h. void abilita();
    i. void visualizzasegn();

## 2.4 Dipendenze tra funzioni

Di seguito un grafico contenente tutte le funzioni e tutte le sottofunzioni dipendenti tra loro.



## 2.5 Flowchart



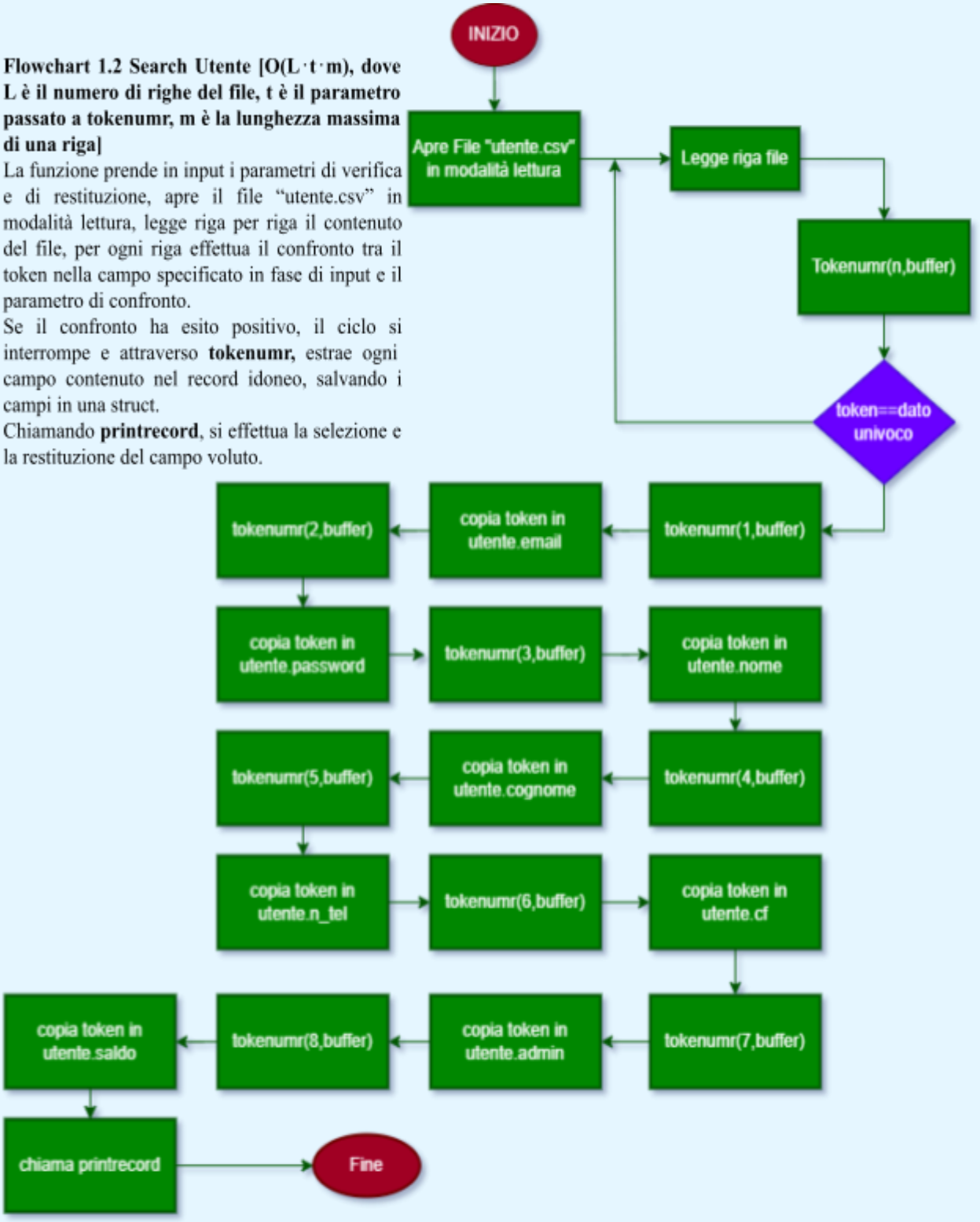

**Flowchart 1.1 Aggiungi carta [O(n)]**
la funzione dichiara le variabili necessarie (numero carta, scadenza(mm/gg),cvv). Fa inserire all'utente i vari dati, ne verifica la lunghezza standard e ricerca, attraverso **searchcarta** che andrà a cercare una corrispondenza del numero della carta all'interno del file "utente.csv".
Qualora la carta inserita dall'utente non fosse già inserita (cardinalità 1:1), si procederà con la chiamata della funzione **inserisci_carta[O(n)]**, la quale inserirà i vari dati all'interno del file "carte.csv" col relativo codice fiscale dell'utente (FK).
Se invece la carta fosse già inserita all'interno del file e/o i dati forniti non rispettassero la lunghezza standard, la funzione darà in output l'avviso relativo all'errore.

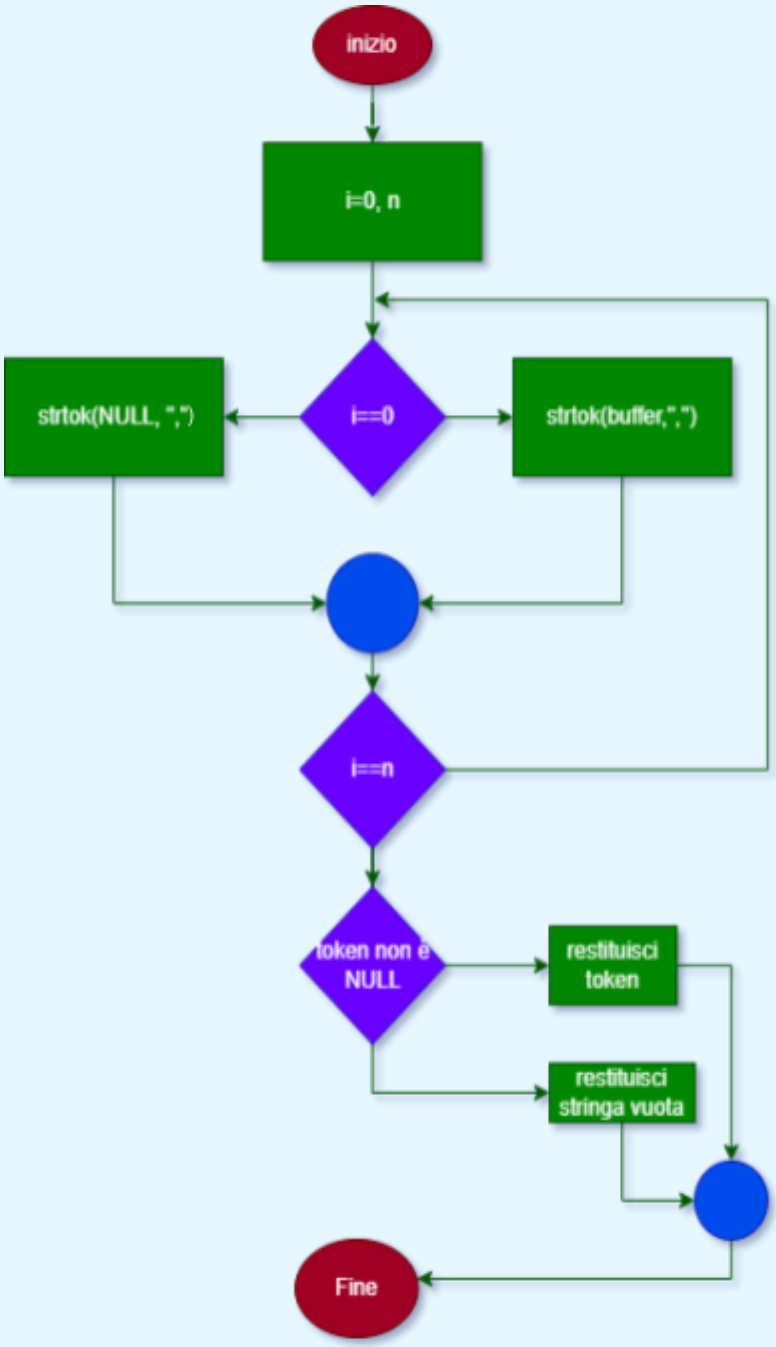
**Flowchart 1.2 Search Utente [O(L·t·m), dove L è il numero di righe del file, t è il parametro passato a tokenumr, m è la lunghezza massima di una riga]**

La funzione prende in input i parametri di verifica e di restituzione, apre il file "utente.csv" in modalità lettura, legge riga per riga il contenuto del file, per ogni riga effettua il confronto tra il token nella campo specificato in fase di input e il parametro di confronto.

Se il confronto ha esito positivo, il ciclo si interrompe e attraverso **tokenumr,** estrae ogni campo contenuto nel record idoneo, salvando i campi in una struct.

Chiamando **printrecord**, si effettua la selezione e la restituzione del campo voluto.



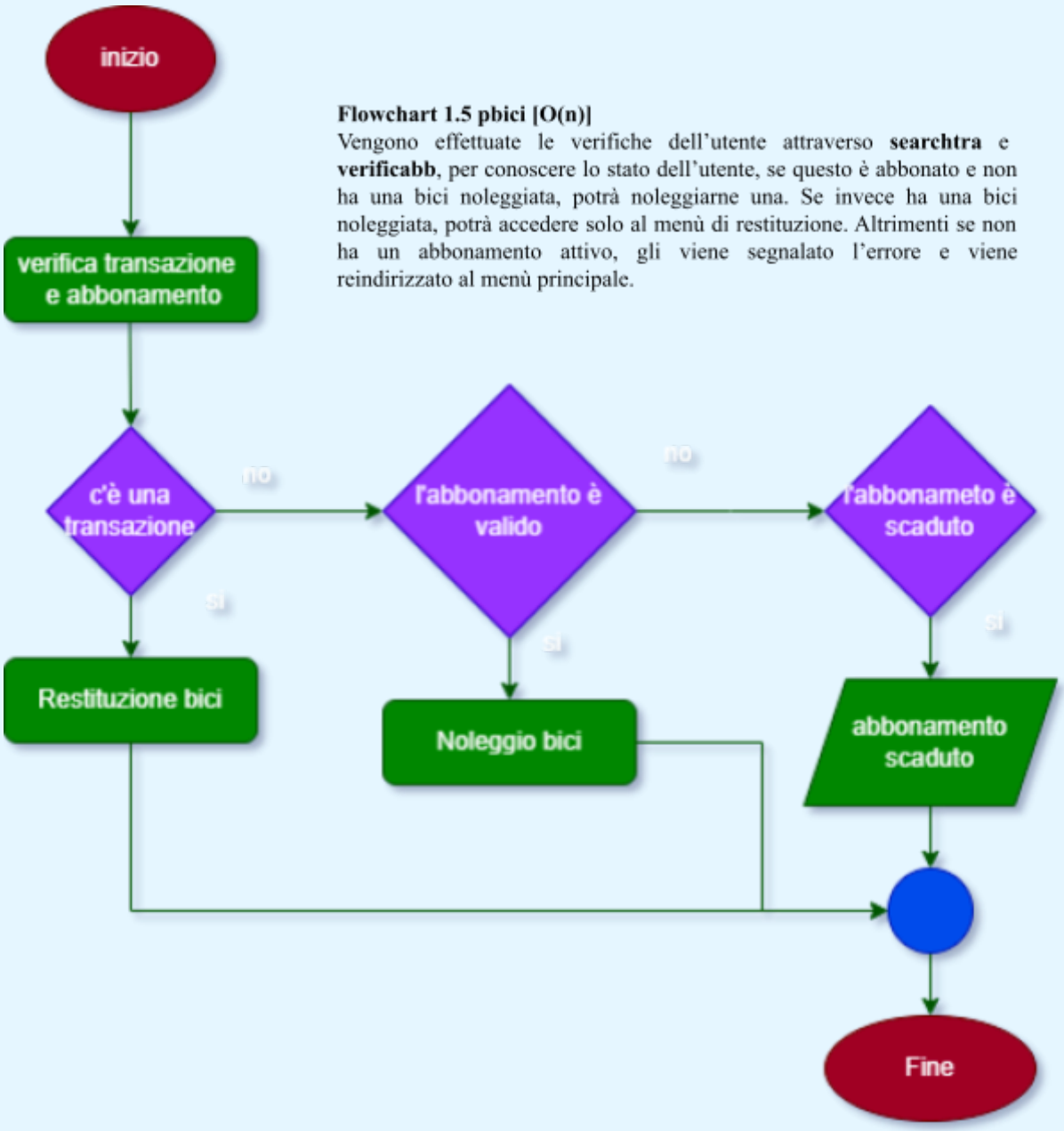



**Flowchart 1.3 Tokenumr [O(t·m), dove t è il numero del campo imposto da chiamata e m è la lunghezza massima].**

Prende in input la posizione del campo da ricercare all'interno della riga (t), attraverso un for itera t-volte, fino ad arrivare alla posizione richiesta.

Per ogni iterazione viene verificato che il counter i sia maggiore di 0, in modo da non impostare nuovamente la riga da cui strtok estrarrà il token.

Strtok è una funzione presente nella libreria string.h, che permette di estrarre, da una stringa, porzioni di testo limitato da dei valori variabili, nel nostro caso, dato che la stringa da cui estrarre i token è una stringa proveniente da un file csv, il limitatore dello strtok sarà la virgola.




**Flowchart 1.5 pbici [O(n)]**
Vengono effettuate le verifiche dell'utente attraverso **searchtra** e **verificabb**, per conoscere lo stato dell'utente, se questo è abbonato e non ha una bici noleggiata, potrà noleggiarne una. Se invece ha una bici noleggiata, potrà accedere solo al menù di restituzione. Altrimenti se non ha un abbonamento attivo, gli viene segnalato l'errore e viene reindirizzato al menù principale.


**Flowchart 1. Create_Complex_Json[O(n)]**

La funzione chiama **readFileContent** che salva in una stringa tutto il contenuto di un file, in questo caso, la prima volta salva tutti gli header e i parametri del file HTML per il contenuto dell'email, la seconda volta legge il contenuto di un file "chiusura" del file HTML precedente, che comprende le varie chiusure dei tag.

Una volta letti i file html e salvati in 2 stringhe differenti, attraverso lo **sprintf** (che serve per concatenare più dati in una stringa) presente nella libreria string.h, viene concatenata la stringa contenente il primo file html con il codice verifica casuale, accodando a questi la stringa del file html di chiusura.

La funzione crea i vari oggetti a partire dal root, ossia l'elemento principale che contiene tutti gli altri elementi all'interno del file json.

Successivamente crea gli oggetti mittente e destinatario, aggiunge il contenuto della stringa dell'HTML nell'oggetto message_version,

definisce tutti gli oggetti nel root, converte il payload in una stringa e chiama **inviaEmail**, la quale imposterà i parametri per la richiesta HTTPS all'API di brevo, in modo tale da inviare l'email all'utente.


**Flowchart CloseTransazione [O(L · t · m), dove L è il numero di righe del file, t è il parametro passato a tokenumr, m è la lunghezza massima di una riga]**

La funzione apre il file "transazione.csv", legge riga per riga per trovare la transazione a cui il programma si riferisce confrontando il campo id transazione e il campo utilizzo (che indica se l'utente ha restituito o meno la bici).

Una volta trovata, attraverso **fseek,** sposta il cursore all'inizio del campo di utilizzo e imposta quest'ultimo a 0, in modo tale da "chiudere" la transazione e archiviarla in quelle passate.

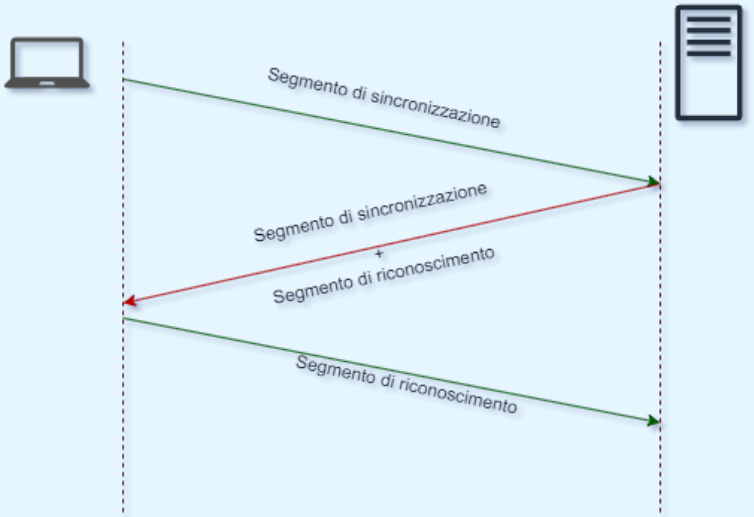
# 3. Codifica

Il ciclo di vita del programma si compone di 3 fasi: accesso, menu principale e menu admin.
Il software è stato sviluppato tramite l'utilizzo delle librerie standard c, e le librerie esterne cJson e CURL. (paragrafo 2.3)

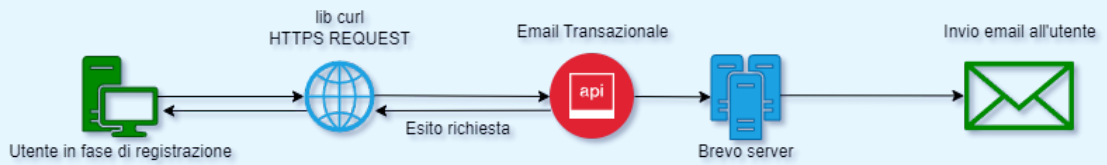
## 3.1 API Brevo

Nell'accesso è stata implementata un'autenticazione a due fattori, che consiste nell'inviare un codice di verifica all'email dell'utente per ogni accesso, attraverso il protocollo SMTP e l'API Brevo.

Attraverso la libreria esterna cJson, è stata implementata un'automazione che crea un payLoad definito in base all'email passata.
Definito il payLoad, questo viene passato alla funzione **inviaEmail** la quale, attraverso la libreria CURL, definisce e imposta i parametri necessari alla richiesta da effettuarsi all'API di Brevo, come ad esempio il metodo di trasmissione POST e il protocollo HTTPS.

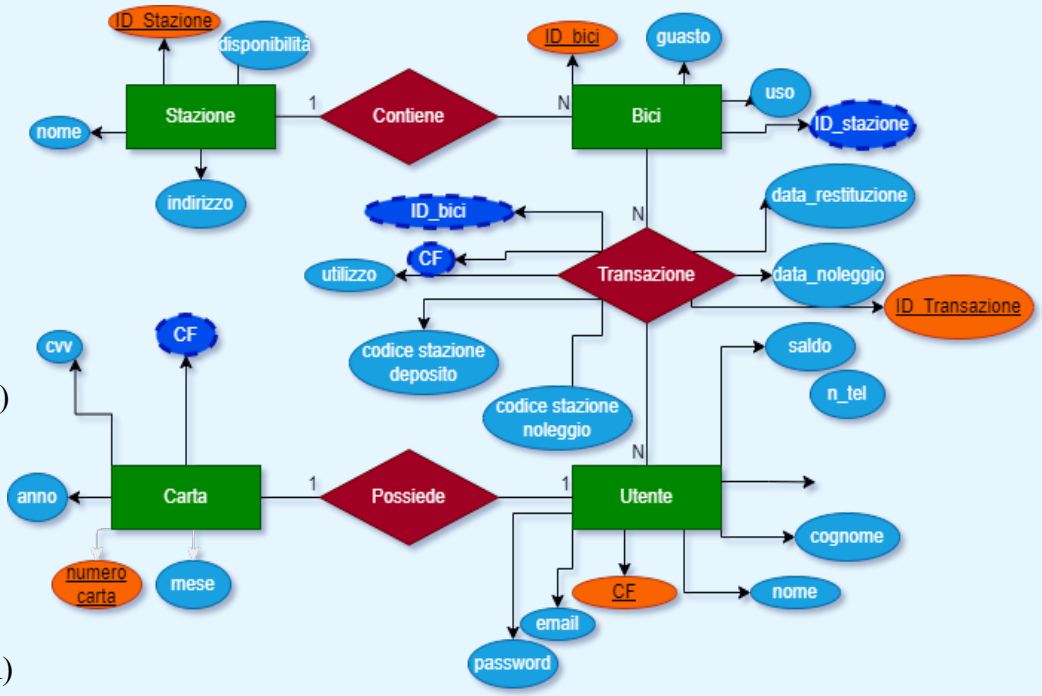



Infine l'API inoltra al client l'esito della richiesta; la comunicazione tra client e server avviene attraverso il protocollo TCP che, attraverso il Three Way Handshaking, assicura il corretto indirizzamento dei pacchetti dati.



## 3.2 Diagramma E/R

Tutta la gestione dei dati è eseguita attraverso un'emulazione di un database relazionale e di query SQL. Ogni tabella di un eventuale database è riportata attraverso dei file csv, all'interno dei quali è possibile definire i vari campi con le varie chiavi primarie ed esterne, garantendo l'integrità referenziale.
Le tabelle sono così definite:




1. **Utente:**
   a. email
   b. password
   c. CF (PK)
   d. nome
   e. Cognome
   f. n_tel
   g. saldo
2. **Carta**
   a. n_carta(PK)
   b. mese
   c. anno
   d. cvv
   e. CF(FK)
3. **Bici**
   a. ID_bici(PK)
   b. guasto
   c. uso
   d. ID_stazione(FK)
4. **Transazione**
   a. ID_transazione(PK)
   b. data_noleggio
   c. data_restituzione
   d. utilizzo
   e. codice_stazione_noleggio
   f. codice_stazione_deposito
   g. ID_bici(FK)
   h. CF(FK)
5. **Stazione**
   a. ID_stazione(PK)
   b. nome
   c. indirizzo
   d. disponibilità

## 3.3 Gestione transazioni bici

In base all'abbonamento esistente, il sistema fornisce un indirizzamento automatico alle pagine di noleggio e deposito, verificando che l'utente non abbia bici noleggiate e che sia abbonato.

Come si evince dai flowchart, la modifica dei dati avviene tramite sovrascrizione del dato da modificare attraverso lo spostamento del cursore di scrittura, nello specifico si utilizza fseek: si calcola la differenza tra la posizione corrente del cursore e la lunghezza dei vari dati successivo quello da sovrascrivere sommata al numero di virgole, secondo la formula seguente.

$$p = pa - \sum_{i=0}^{k} (l + v)_i$$

Dove p è il numero di settori di avanzamento, k è il numero totale di campi successivi, pa è la posizione attuale del file, $(l + v)$ è la somma del numero dei caratteri di una stringa successivo ($l$) sommato alla suo divisore di campo ($v$), ossia la virgola.

Per ovviare ad eventuali errori di scrittura, per tutti i dati modificabili è stato impostato un limite massimo di caratteri utilizzati, per il quale ogni volta che il nuovo dato viene sovrascritto, la sua lunghezza non può eccedere quella del precedente.

Ogni noleggio è salvato come transazione aperta, in questa vengono salvate le date di noleggio e di scadenza, l'id della stazione di noleggio, il codice fiscale dell'utente e l'id della bici noleggiata, viene aggiunto un campo booleano "uso" che simboleggia lo stato della transazione e lo si imposta a 1 (aperta).

Al momento della restituzione della bici il sistema trova la transazione aperta dall'utente e aggiunge ad essa l'id della stazione del deposito; successivamente viene modificato lo stato della transazione impostandolo a 0 (chiusa).

```
Scadenza abbonamento tra 28 giorni.
-------------------------------------------------------------------
|Stazioni:      |       Indirizzo:        | Disponibilità: | Codice: |
-------------------------------------------------------------------
|Ebalia         |P.za Ebalia              |05              |11 |
-------------------------------------------------------------------
|Consiglio      |Via Consiglio Park       |13              |12 |
-------------------------------------------------------------------
|Dante          |P.zle Bestat 5           |12              |13 |
-------------------------------------------------------------------
|d'Aquino       |Via d'Aquino 9           |10              |14 |
-------------------------------------------------------------------
|Liguria        |Via Liguria 46           |10              |15 |
-------------------------------------------------------------------
|Aristosseno    |Via aristosseno 35       |10              |16 |
-------------------------------------------------------------------
|Peripato       |Via Viola                |10              |17 |
-------------------------------------------------------------------
|Duomo          |Via Duomo 140            |10              |18 |
-------------------------------------------------------------------
|Martilotti     |Via Orsini 122           |10              |19 |
-------------------------------------------------------------------
|T.Centrale     |P.za della Libertà      |10              |20 |
-------------------------------------------------------------------
|DeGasperi      |Via de Gasperi 1         |10              |21 |
-------------------------------------------------------------------
|Cannata        |Via Cannata              |09              |22 |
-------------------------------------------------------------------
|L.Como         |Via L.di Como 67         |10              |23 |
-------------------------------------------------------------------
|L.Lugano       |Via L.di Lugano 13       |10              |24 |
-------------------------------------------------------------------
|Litoranea      |Viale dei Mioperi Pulsano|12              |25 |
-------------------------------------------------------------------
|DiGrassi       |P.za Restist. Conversano |08              |26 |
-------------------------------------------------------------------
|Schiavo        |Via Petrarca 39 Crispiano|10              |27 |

Inserisci il codice della stazione:
11


Hai 3h di bici libere, vuoi prenotarla?
1) Si
2) No
1
Data abbonamento: 08:07:25  09/06/2024Transazione effettuata con successo!
Id Transazione:105
Id Bici: 115
Data Noleggio: 08:07:25  09/06/2024
Data Scadenza: 11:07:25  09/06/2024

Riporta la bici in una stazione per non incorrere in una sanzione!
Grazie per aver scelto EcoRide
```

## 3.4 Test

| Codice requisito | Codice test | Nome | Descrizione test | Eventuale input | Risultato atteso | Risultato ottenuto |
|---|---|---|---|---|---|---|
| R00 | 0.1 | main | avvio del programma | | il menu utente viene aperto all'utente | il menu utente viene aperto all'utente |
| R00 | 0.2 | main | avvio del programma | | il menu admin viene aperto all' admin | il menu admin viene aperto all' admin |
| R01 | 1.1 | caricamento | avvio del programma | | il programma stampa una barra di caricamento | il programma stampa una barra di caricamento |
| R02 | 2.1 | printlogo | avvio del programma | | stampa logo ecoride | stampa logo Ecoride |
| R03 | 3.1 | printwelcome | avvio del programma | | stampa benvenuto | stampa benvenuto |
| R04 | 4.1 | login | l'utente inserisce i propri dati per il login | s.digrassi@studenti.uniba.it<br>1234 | credenziali utente corrette, quindi accesso consentito | accesso consentito |
| R04 | 4.2 | login | l'utente inserisce i dati sbagliati per il login | s.digrassi@studenti.uniba.it<br>0123 | credenziali utente sbagliate, quindi accesso negato | stampa messaggio di errore |

| | | | | | | |
|---|---|---|---|---|---|---|
| R05 | 5.1 | verifica | verifica con credenziali corrette | s.digrassi@studenti.uniba.it<br>1234 | restituisce 1 | restituisce 1 |
| R05 | 5.2 | verifica | verifica con credenziali errate | s.digrassi@studenti.uniba.it<br>0123 | restituisce 0 | restituisce 0 |
| R06 | 6.1 | searchutente | ricerca email da cf corretto | MRTNDR04H15L049D | campo email dal record utente trovato | campo email dal record utente trovato |
| R06 | 6.2 | searchutente | ricerca email da cf errato | MRTNDR01 | record utente non trovato | restituisce NULL |
| R06 | 6.3 | searchutente | ricerca password da cf corretto | MRTNDR04H15L049D | campo password dal record utente trovato | campo password dal record utente trovato |
| R06 | 6.3 | searchutente | ricerca password da cf errato | MRTNDR04H15 | record utente non trovato | restituisce NULL |
| R06 | 6. | searchutente | ricerca saldo da cf corretto | MRTNDR04H15L049D | campo saldo da record utente trovato | record utente trovato |
| R06 | 6. | searchutente | ricerca saldo da cf errato | MRTNDR04 | record utente non trovato | restituisce NULL |
| R06 | 6. | searchutente | ricerca admin da cf corretto | MRTNDR04H15L049D | campo admin da record utente trovato | record utente trovato |
| R06 | 6. | searchutente | ricerca admin da cf errato | MRTNDR04 | record utente non trovato | restituisce NULL |

| | | | | | | |
|---|---|---|---|---|---|---|
| R07 | 7.1 | reg | tentativo reg con cf non registrato | MRTNDR 0 | record utente non trovato | l'utente effettua la registrazione |
| R07 | 7.2 | reg | tentativo reg con cf già registrato | MRTNDR 04H15L04 9D | record utente trovato | la registrazione non viene effettuata |
| R07 | 7.3 | reg | tentativo reg con email non registrata | m.schiavo2 @studenti. uniba.it | record utente non trovato | la registrazione viene effettuata |
| R07 | 7.4 | reg | tentativo reg con email registrata | s.digrassi@ studenti.uni ba.it | record utente trovato | la registrazione non viene effettuata |
| R07 | 7.5 | reg | tentativo reg con codice verifica email esatto | 265896 | la registrazione viene effettuata | la registrazione viene effettuata |
| R07 | 7.6 | reg | tentativo reg con codice verifica email errato | 265897 | la registrazione non viene effettuata | la registrazione non viene effettuata |
| R08 | 8.1 | verifica_e mail | tentativo di invio email a un indirizzo corretto | m.schiavo2 @studenti. uniba.it | l'email viene inviata | l'email viene inviata |
| R08 | 8.2 | verifica_e mail | tentativo di invio email a un indirizzo inesistente | m.schiavo2 studenti.uni ba.it | curl stampa l'errore | curl stampa l'errore |
| R09 | 9.1 | create_co mplex_jso n | vengono creati i vari oggetti json per le impostazioni dei parametri per gestire l'invio dell'email | | oggetti Json creati | oggetti Json creati |

| | | | | | | |
|---|---|---|---|---|---|---|
| R10 | 10.1 | readFileContent | lettura di un file presente nella directory | “ecohtml1.html” | il contenuto del file viene convertito in stringa e passato al software | il contenuto del file viene convertito in stringa e passato al software |
| R10 | 10.2 | readFileContent | tentativo di lettura di un file non presente in directory | “ecophp1.php” | il contenuto del file non viene convertito in stringa | il contenuto del file non viene convertito in stringa |
| R11 | 11.1 | inviaEmail | tentativo di registrazione con email valida | “a.martilotti2@studenti.uniba.it” | la funzione imposta i parametri per la richiesta https all’ API di brevo e invia l’email | la funzione imposta i parametri per la richiesta https all’ API di brevo e invia l’ email |
| R11 | 11.2 | inviaEmail | tentativo di registrazione con email inesistente | “amartilotti2studenti.uniba.it” | non viene inviata alcuna email e il programma restituisce un messaggio di errore | non viene inviata alcuna email e il programma restituisce un messaggio di errore |
| R12 | 12.1 | randoma | generazione codice casuale per verifica email | | viene generato un codice randomico di 5 cifre | viene generato un codice randomico di 5 cifre |
| R13 | 13.1 | paginaabbonamento | utente decide di visualizzare il menu abbonamento | | viene visualizzata la pagina di gestione dell’ abbonamento | viene visualizzata la pagina di menu dell’ abbonamento |

| | | | | | | |
|---|---|---|---|---|---|---|
| R14 | 14.1 | visabbonamento | visualizzazione dei dettagli dell'abbonamento | input codice fiscale utente | dettagli abbonamento visualizzati | dettagli abbonamento visualizzati |
| R14 | 14.2 | visabbonamento | visualizzazione dei dettagli dell'abbonamento di un utente non abbonato | | dettagli abbonamento non visualizzati | abbonamento non trovato, stampa solamente il saldo |
| R15 | 15.1 | searchs | ricerca del saldo | input codice fiscale utente | viene trovato il saldo dell'account dell'utente | viene trovato il saldo dell'account dell'utente |
| R15 | 15.2 | searchs | ricerca del saldo | input codice fiscale utente errato | non viene trovato il saldo dell'account dell'utente | non viene trovato il saldo dell'account dell'utente |
| R16 | 16.1 | rinnova | l'utente può scegliere tra abbonamento mensile e settimanale | | l'utente ha scelto la tipologia di abbonamento, ha rinnovato l'abbonamento | l'utente ha scelto la tipologia di abbonamento, ha rinnovato l'abbonamento |
| R16 | 16.2 | rinnova | l'utente è già abbonato | | l'utente non ha potuto scegliere la tipologia di abbonamento | l'utente non ha potuto scegliere la tipologia di abbonamento |
| R17 | 17.1 | modificasaldo | l'utente ha scelto un abbonamento e le spese per lo stesso vengono sottratte dal saldo | | il saldo viene aggiornato | il saldo viene aggiornato |

| | | | | | | |
|---|---|---|---|---|---|---|
| R17 | 17.2 | modificasaldo | l'utente ha già un abbonamento in corso | | il saldo non viene aggiornato | il saldo non viene aggiornato |
| R18 | 18.1 | cov | calcolo scadenza e durata residua dell' abbonamento | | viene calcolata la durata residua dell' abbonamento | viene calcolata la durata residua dell' abbonamento |
| R18 | 18.2 | cov | calcolo scadenza con data nel formato errato | | non viene calcolata la durata residua dell'abbonamento | non viene calcolata la durata residua dell'abbonamento |
| R19 | 19.1 | verificaabb | verifica che l'utente sia o meno abbonamento | l'utente sta rinnovando l'abbonamento | viene verificata la validità dell' abbonamento | viene verificata la validità dell'abbonamento |
| R19 | 19.2 | verificaabb | verifica che l'utente sia abbonato o meno, l'utente non è abbonato | l'utente sta rinnovando l'abbonamento | stampa avviso, abbonamento già effettuato | stampa avviso, abbonamento già effettuato |
| R20 | 20.1 | paginacarta | scelta tra pagina di modifica o aggiungi carta | l' utente può scegliere se aggiungere o modificare carta | l'utente accede alla pagina di modifica o aggiunta carta | l'utente accede alla pagina di modifica o aggiunta carta |

| | | | | | | |
|---|---|---|---|---|---|---|
| R21 | 21.1 | aggiungicarta | l'utente inserisce un nuovo metodo di pagamento valido | 5478965896321256<br>12<br>25<br>658 | ha verificato che non siano già presenti i dati che ha raccolto e che siano conformi allo standard di lunghezza | ha verificato che non siano già presenti i dati che ha raccolto e che siano conformi allo standard di lunghezza |
| R21 | 21.2 | aggiungicarta | l'utente inserisce un nuovo metodo di pagamento errato | 5412563<br>2<br>3333<br>33 | ha verificato che non siano già presenti i dati che ha raccolto e che siano conformi allo standard di lunghezza | non viene verificata l'eventuale presenza dei dati di pagamento, né la loro conformità agli standard di lunghezza |
| R22 | 22.1 | inscard | inserimento nuovi dati sul file "carte.csv" | 5478965896321256<br>12<br>25<br>658 | sul file "carte.csv" è stata creata una nuova riga con i nuovi dati dell' utente | sul file "carte.csv" è stata creata una nuova riga con i nuovi dati dell' utente |
| R23 | 23.1 | modifica | l'utente sta modificando i dati del suo metodo di pagamento | 5333333333333333<br>33<br>33<br>333 | la riga individuata viene modificata | la riga individuata viene modificata |

| | | | | | | |
|---|---|---|---|---|---|---|
| R23 | 23.2 | modifica | l'utente sta modificando i dati del suo metodo di pagamento che non rispettano gli standard di lunghezza | 533333333 3333333<br>3333<br>33<br>333 | la riga viene individuata, ma non vengono modificati i dati | la riga viene individuata, ma non vengono modificati i dati |
| R24 | 24.1 | aperturamappa | l'utente apre la pagina delle mappe | | la pagina web viene aperta tramite browser | la pagina web viene aperta tramite browser |
| R24 | 24.2 | apperturamappa | l'utente apre la pagina delle mappe, ma non è connesso a internet | | la pagina web non viene aperta tramite browser | non viene aperta alcuna pagina web |
| R25 | 25.1 | pagina_bici | l'utente abbonato non possiede ha transazioni aperte | | l'utente viene reindirizzato alla pagina di noleggio | l'utente viene reindirizzato alla pagina di noleggio |
| R25 | 25.2 | pagina_bici | l'utente ha una transazione aperta | | l'utente viene reindirizzato alla pagina di restituzione | l'utente viene reindirizzato alla pagina di restituzione |
| R26 | 26.1 | verificatrans | l'utente non ha già una transazione aperta | | la verifica sulle eventuali transazioni aperte dell' utente giunge al termine | la verifica sulle eventuali transazioni aperte dell'utente giunge al termine |

| | | | | | | |
|---|---|---|---|---|---|---|
| R26 | 26.2 | verificatrans | l'utente ha già una transazione aperta | | la verifica sulle eventuali transazioni aperte dell' utente giunge al termine | la verifica sulle eventuali transazioni aperte dell' utente non giunge a termine. |
| R27 | 27.1 | noleggio | l'utente inserisce idstazione, corrispondente alla stazione da lui scelta | 11<id stazione<n stazioni | viene verificata la disponibilità della stazione, corrispondente a tale id | viene verificata la disponibilità della stazione, corrispondente a tale id |
| R27 | 27.2 | noleggio | l'utente inserisce idstazione, non corrispondente alla stazione da lui scelta | 11<nstazioni<id_stazione | viene verificata la disponibilità della stazione, corrispondente a tale id | l'id della stazione non viene acquisito e quindi non viene verificata la disponibilità della stazione |
| R28 | 28.1 | verifica_disp_stazione | calcolo stazione con codice corretto | 11<id stazione<n stazioni | viene verificata la disponibilità di almeno 1 bici per la stazione in input | viene verificata la disponibilità di almeno 1 bici per la stazione in input |
| R28 | 28.2 | verifica_disp_stazione | calcolo stazione con codice errato | 11<nstazioni<id_stazione | non viene verificata la disponibilità di almeno 1 bici per la stazione in input | non viene verificata la disponibilità di almeno 1 bici per la stazione in input |

| | | | | | | |
|---|---|---|---|---|---|---|
| R29 | 29.1 | opbici | tentativo di noleggio bici con codice corretto | 11<id stazione<n stazioni | trova la bici disponibile in una determinata stazione, ne controlla eventuali guasti. Avvia il processo di registrazione della bici come “NOLEGGIATA”. Imposta “NON DISPONIBILE” come stato della bici | trova la bici disponibile in una determinata stazione, ne controlla eventuali guasti. Avvia il processo di registrazione della bici come “NOLEGGIATA”. Imposta “NON DISPONIBILE” come stato della bici |
| R29 | 29.2 | opbici | tentativo di noleggio bici con codice errato | 11<nstazioni<id stazione | trova la bici disponibile in una determinata stazione, ne controlla eventuali guasti. Avvia il processo di registrazione della bici come “NOLEGGIATA”. Imposta “NON DISPONIBILE” come stato della bici | non trova bici disponibili e non imposta alcuno stato della bici |
| R30 | 30.1 | opentransazione | viene noleggiata la bici e si crea la transazione | n | viene calcolata la data e l’ora di noleggio e di scadenza, viene salvato su file lo stato della | viene calcolata la data e l’ora di noleggio e di scadenza, viene salvato su file lo stato della |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | | | | transazione a “TRUE”. Invia all’utente le informazioni riguardanti il noleggio | transazione a “TRUE”. Invia all’utente le informazioni riguardanti il noleggio |
| R30 | 30.2 | opentransazione | la bici non è disponibile | n+k | viene calcolata la data e l’ora di noleggio e di scadenza, viene salvato su file lo stato della transazione a “TRUE”. Invia all’utente le informazioni riguardanti il noleggio | non viene calcolata alcuna scadenza, né viene salvato su file lo stato della transazione. non invia all’ utente le informazioni riguardanti il noleggio |
| R31 | 31.1 | decrementa_stazione | viene passato un dato compreso tra 11 e n | 11<dato<nstazione | la disponibilità della stazione viene decrementata di 1 | la disponibilità della stazione viene decrementata di 1 |
| R31 | 31.2 | decrementa_stazione | viene passato un dato che non rispetta l’intervallo | 11<nstazioni<dato | la disponibilità della stazione viene decrementata di 1 | la disponibilità della stazione rimane la medesima |
| R32 | 32.1 | stampastazioni | vengono stampate le stazione | | le stazioni vengono stampate | le stazioni vengono stampate |
| R32 | 32.2 | stampastazioni | il file “stazioni.csv” non è presente in directory | | le stazioni non vengono stampate nella tabella | non viene stampata la tabella delle stazioni |

| | | | | | | |
|---|---|---|---|---|---|---|
| R33 | 33.1 | calcolastazione | vengono calcolate le stazioni presenti | | viene calcolato il numero delle stazioni disponibili | viene calcolato il numero delle stazioni disponibili |
| R34 | 34.1 | restituzione | l'utente ha già una bici | | l'utente ha già noleggiato una bici e viene reindirizzato | l'utente ha già noleggiato una bici e viene reindirizzato |
| R35 | 35.1 | segnalazione | l'utente ha già una bici e vuole segnalare un problema | | la descrizione della segnalazione viene salvata nel file "segnalazioni.csv" | la descrizione della segnalazione viene salvata nel file "segnalazioni.csv" |
| R36 | 36.1 | incrementa_bici | incrementa bici con id n corretto | idbici | la disponibilità della bici viene modificata nel file "bici.csv" | la disponibilità della bici viene modificata nel file "bici.csv" |
| R36 | 36.2 | incrementa_bici | incrementa bici con id n errato | idbici+e | la disponibilità della bici non viene modificata nel file "bici.csv" | la bici non è disponibile |
| R37 | 37.1 | closetransazione | ricerca transazione t, a partire dal codice fiscale corretto e chiusura | MRTNDR04H15L049D | il campo "utilizzo" di "transazione.csv" relativo a quella bici, viene impostato a 0 | il campo "utilizzo" di "transazione.csv" relativo a quella bici, viene impostato a 0 |

| | | | | | | |
|---|---|---|---|---|---|---|
| R37 | 37.2 | closetransazione | ricerca transazione t, a partire dal codice fiscale errato e tentativo chiusura | MRTND | non viene trovato nulla | non viene trovato nulla |
| R38 | 38.1 | incrementa_stazione | incremento stazione | idstazione | incremento stazione n effettuato | incremento stazione n effettuato |
| R39 | 39.1 | tokenumr | viene estrapolato il token n di una stringa n+k | n | campo richiesto trovato | campo richiesto trovato |
| R39 | 39.2 | tokenumr | viene estrapolato il token n di una stringa n-k | n | campo richiesto non trovato | campo richiesto non trovato |
| R40 | 40.1 | printrecord | viene chiesto il dato n | n | separa i vari campi della struct e restituisce il campo n | separa i vari campi della struct e restituisce il campo n |
| R41 | 41.1 | searchcarta | ricerca campo t in carta.csv a partire dal cf corretto | MRTNDR04H15L049D | campo richiesto trovato | campo richiesto trovato |
| R41 | 41.2 | searchcarta | ricerca campo t in carta.csv a partire dal cf errato | MRTNDR0 | campo richiesto non trovato | campo richiesto non trovato |
| R42 | 42.1 | printcarta | viene chiesto il dato n della struct carta | struct carta | restituisce il contenuto relativo alla carta, in base al campo richiesto | restituisce il contenuto relativo alla carta, in base al campo richiesto |

| | | | | | | |
|---|---|---|---|---|---|---|
| R43 | 43.1 | ecosearchtra | ricerca campo transazione da cf | MRTNDR04H15L049D | viene trovato un determinato campo di una transazione in base al dato univoco | viene trovato un determinato campo di una transazione in base al dato univoco. |
| R43 | 43.2 | ecosearchtra | ricerca campo transazione da cf errato | MRTND | non viene trovato un determinato campo di una transazione | non viene trovato alcun campo |
| R44 | 44.1 | ecoFindIDSt | ricerca la stazione di noleggio di una transazione a partire dal suo id | 32 | trovato l'id di una stazione a partire dall' id della transazione | trovato l'id di una stazione a partire dall' id della transazione |
| R44 | 44.2 | ecoFindIDSt | ricerca l'id stazione di noleggio di una transazione a partire dal suo id | -3 | non viene trovato l' id di una stazione a partire dall' id della transazione | non viene trovato nessun id |
| R45 | 45.1 | gestioneadmin | l'admin ha deciso di controllare gli admin | 1 | stampa il menu con le varie operazioni admin | stampa il menu con le varie operazioni admin |

| | | | | | | |
|---|---|---|---|---|---|---|
| R45 | 45.2 | gestionead min | l'utente non è un admin | 0 | non viene stampato il menu con le varie operazioni admin | non viene stampato il menu admin |
| R46 | 46.1 | impostaad min | l'utente è un admin | 0 | il campo admin dell' utente viene impostato a 1 | il campo admin dell' utente viene impostato a 1 |
| R46 | 46.2 | impostaad min | l' admin sta impostando un altro utente come admin | 1 | il campo admin dell' utente viene impostato a 1 | il campo admin dell' utente rimane invariato |
| R47 | 47.1 | rimuoviad min | l' admin sta cercando di rimuovere un admin con il codice fiscale corretto | MRTNDR 04H15 | l' admin viene trovato e il suo campo "admin" viene portato a 0 | l' admin viene trovato e il suo campo "admin" viene portato a 0 |
| R47 | 47.2 | rimuoviad min | l' admin sta cercando di rimuovere un admin con il codice fiscale errato | MRTYT | l' admin viene trovato e il suo campo "admin" viene portato a 0 | l' admin non viene trovato |
| R48 | 48.1 | visualizzaa dmin | vengono stampati tutti gli admin con campo impostato a 1 | | tutti gli utenti, aventi il campo admin impostato a 1, vengono stampati | tutti gli utenti, aventi il campo admin impostato a 1, vengono stampati |

| | | | | | | |
|---|---|---|---|---|---|---|
| R48 | 48.2 | visualizzaadmin | vengono stampati tutti gli admin con campo impostato a 0 | | tutti gli utenti, aventi il campo admin impostato a 1, vengono stampati | non viene stampato nulla |
| R49 | 49.1 | gestionebici | visualizzazione menu delle operazione sulle bici | | il menu con le varie operazioni sulle bici viene visualizzato | il menu con le varie operazioni sulle bici viene visualizzato |
| R50 | 50.1 | inseriscibici | inserimento nuova bici con codice corretto | codice stazione 20 | viene inserita una nuova bici | viene inserita una nuova bici |
| R50 | 50.2 | inseriscibici | inserimento nuova bici con codice errato | codice stazione 30 | viene inserita una nuova bici | non viene inserita la bici |
| R51 | 51.1 | visualizza Bici | elenco delle bici con id corretto | | viene stampato l' elenco delle bici | viene stampato l' elenco delle bici |
| R52 | 52.1 | visualizza BiciUso | stampa bici in uso | | tutte le bici in uso vengono stampate | tutte le bici in uso vengono stampate |
| R52 | 52.2 | visualizza BiciUso | stampa bici in uso | | tutte le bici in uso vengono stampate | non viene stampato nulla |
| R53 | 53.1 | gestionefeedback | stampa dei feedback | l' utente inserisce il proprio feedback | vengono stampati i feedback in una tabella | vengono stampati i feedback in una tabella |

| | | | | | | |
|---|---|---|---|---|---|---|
| R54 | 54.1 | gestionesegnalazioni | stampa segnalazioni | l' utente inserisce una segnalazioni | viene stampato un menu contenente le varie operazioni | viene stampato un menu contenente le varie operazioni |
| R55 | 55.1 | disabilita | l'admin disabilita una bici guasta | id_bici | la bici viene impostata come guasta quindi il campo verrà modificato a 1 | la bici viene impostata come guasta quindi il campo verrà modificato a 1 |
| R55 | 55.2 | disabilita | l'admin abilita una bici già guasta | id_bici | la bici viene impostata come guasta quindi il campo resterà invariato a 1 | la bici guasta rimarrà impostata come funzionante |
| R56 | 56.1 | abilita | l'admin abilita una bici non più guasta | id_bici | la bici impostata come guasta, viene reimpostata a funzionante modificando il campo a 0 | la bici impostata come guasta, viene reimpostata a funzionante modificando il campo a 0 |
| R56 | 56.2 | abilita | l'admin abilita una bici non più guasta | id_bici+e | la bici impostata come guasta, viene reimpostata a funzionante modificando il campo a 0 | non vengono apportate modifiche alla bici impostata come guasta |

| | | | | | | |
|---|---|---|---|---|---|---|
| R57 | 57.1 | ins | inserisci i dati della nuova stazione | id_stazione | il dato verrà inserito nel file “stazioni.csv ” | il dato verrà inserito nel file “stazioni.csv ” |
| R58 | 58.1 | gestionestazioni | inserimento dei dati della nuova stazione in “stazione.csv” | id_stazione | i dati relativi a una nuova stazione saranno inseriti in “stazione.csv ” | i dati relativi a una nuova stazione saranno inseriti in “stazione.csv ” |
| R59 | 59.1 | gestionetrans | viene stampato una lista delle transazioni | | stampa una lista con tutte le transazioni ordinate in base al codice stazione | stampa una lista con tutte le transazioni ordinate in base al codice stazione |
| R59 | 59.2 | gestionetrans | stampa una lista con tutte le transazioni ordinate in base al codice stazione errato | | stampa una lista con tutte le transazioni ordinate in base al codice stazione | non vengono stampate le transazioni |
| R60 | 60.1 | printadmin | avvio menu admin | | l’ ascii art viene stampato per il menu admin | l’ ascii art viene stampato per il menu admin |

# 4. Conclusioni

Il software si presenta stabile al lancio (99.7%), non riscontrando bug evidenti.
Sono stati sviluppati tutti i punti richiesti provando a massimizzare l'ottimizzazione della gestione dei dati attraverso query complesse, ma variabili e solide.
Le transazioni sono state impostate in modo intuitivo per l'utente, fornendo tutti i dettagli relativi.

## 4.1 Punti di forza

1. Gestione Dati
2. 2FA
3. Interfaccia intuitiva per l'utente
4. Stabilità
5. Informazione dinamica
6. Monitoraggio admin

## 4.2 Punti di debolezza

1. Compatibilità esclusiva con NixOS
2. Controllo su composizione stringa del codice fiscale (verificando che siano state correttamente inserite 6 lettere + 2 cifre + 1 lettera + 2 cifre + 1 lettera + 3 cifre + 1 lettera)
3. Mancanza di una GUI dedicata
4. Gestione delle segnalazioni dipendente dalla disponibilità dell'admin

## 4.3 Potenziamenti Futuri

1. **Compatibilità**: rendere il software compatibile per i sistemi operativi maggiormente utilizzati.

2. **Database relazionale**: in futuro si potrebbe pensare a un'emigrazione verso un database relazionale di tipo SQL, in modo da poter gestire meglio le operazioni sui dati, mantenere i dati in un server centralizzato e creare nuove funzioni attraverso query SQL (es. MariaDB, HeidiSQL).

3. **Tracciamento GPS**: per aumentare la sicurezza e il controllo delle biciclette, si potrebbe implementare un localizzatore GPS su ogni bicicletta.

4. **Eliminazione dati:** un database SQL renderebbe molto più semplice la gestione dei dati, per cui anche l'eliminazione degli stessi potrebbe essere facilmente implementata.

5. **Interfaccia grafica:** attraverso le libreria GTK o SDL, si potrebbe implementare un'interfaccia grafica intuitiva per l'utente, in modo tale da non presentare più il programma nel terminale.

6. **Password mask:** implementare un algoritmo che mascheri a schermo ogni carattere immesso dall'utente nell'ambito dell'inserimento password.

7. **Crittografia:** si potrebbe implementare l'algoritmo di cifratura a chiave asimmetrica, per cifrare la comunicazione tra client-server, e di conseguenza anche delle password.

8. **Calcolo automatico cf:** ampliando le funzioni di acquisizione dei dati durante la registrazione, acquisendo gli ulteriori dati utili, si potrebbe implementare un algoritmo che calcoli automaticamente il codice fiscale.

9. **Chatbot AI:** inserire un chatbot addestrato per le assistenze semplici, che non implicano la risposta dell'admin.

10. **Chat assistenza utente-admin:** implementare un sistema di messaggistica istantanea tramite il quale l'utente potrà comunicare in via diretta con gli amministratori del sistema, i quali potranno offrirgli assistenza immediata.

11. **Notifica segnalazioni:** implementare attraverso brevo, un'automazione che invii un'email a ogni admin, ogni volta che un utente effettua una segnalazione.

# 5. Considerazioni finali e ringraziamenti

Il progetto EcoRide ha preso vita nel maggio 2024 e i suoi sviluppatori, dediti al progetto, hanno lavorato per portare sul mercato un software innovativo, pratico, funzionale e intuitivo.

Sebbene sia in versione alpha, si prevedono copiose migliorie, per poter offrire agli utenti sempre più funzionalità, al fine di ottimizzare i tempi richiesti per il noleggio o per l'utilizzo generale del programma, sviluppando una portabilità del software sui vari OS utilizzati nel corrente anno, come Android e iOS.

La mission del progetto mirerà costantemente all'incentivazione dell'utilizzo di mezzi di trasporto ecosostenibili che, nel loro piccolo, possano comunque contribuire al mantenimento di un'ottima salute dell'utente finale.

Le considerazioni di carattere universitario volgono al ringraziamento nei confronti del prof. Vessio per aver formulato le tracce in vista dell'esame di Laboratorio di Informatica che, indipendentemente dai risultati, ha creato l'occasione perfetta allo sviluppo di competenze trasversali quali team-building, workgroup, leadership, problem solving, flessibilità, creatività, capacità di gestione del tempo e pensiero critico, permettendo agli allievi di immergersi in un caso di studio reale, che possa in futuro essere oggetto di un reale investimento e implementazione.

Si ringrazia per l'attenzione.

Samuele Di Grassi
Andrea Martilotti
Marika Schiavo

EcoRide