טכנולוגיות אינטרנט מתקדמות - 61776 WEB))

הגשת פרויקט

<ETH Mainnet Wallet Manager> <B8> <12>

|  |  |
| --- | --- |
| שם חבר הצוות | ת"ז |
| ג'וזיף דאמוני | 322465105 |
| ודיע טנוס | 212248314 |
| גורג' אסחאק | 212243307 |
| אמיר מסעוד | 315908269 |
| מוחמד חטיב | 324056316 |

|  |  |  |
| --- | --- | --- |
| **שם חבר הצוות** | **משימות שהוקצו** | **משימות שהושלמו** |
| ג'וזיף דאמוני | תכנון מבנה הנתונים, פיתוח API | הושלם - API נפרס ונבדק |
| ודיע טנוס | עיצוב, בניית ה- Front-End | הושלם - עיצוב + דפי התחברות/דאשבורד |
| גורג' אסחאק | פיתוח מנגנון אימות משתמשים, Firebase | |  | | --- | |  |   הושלם - אימות והתחברות דרך Firebase |
| אמיר מסעוד | חיבור ל- CoinGecko API, הצגת נתוני מטבעות | הושלם - נתונים מוצגים ב-Front-End |
| מוחמד חטיב | תיעוד הפרויקט, בדיקות ותיקוני באגים | הושלם - דוקומנטציה מלאה ובדיקות איכות |

**תקציר הפרויקט - ETH Wallet Manager**

ETH Wallet Manager הוא יישום מתקדם לניהול ארנקי Ethereum, המאפשר למשתמשים ליצור, לשחזר ולנהל ארנקים דיגיטליים בצורה מאובטחת ויעילה. המערכת מספקת מידע בזמן אמת על מחירי מטבעות קריפטוגרפיים, כולל ETH וטוקנים נוספים, באמצעות CoinGecko API, ומאפשרת שליחת וקבלת נכסים דיגיטליים ישירות מהבלוקצ'יין תוך שימוש ב- Ethereum RPC API . בנוסף, המערכת מציגה היסטוריית עסקאות מלאה עם פירוט סכומים, כתובות יעד וסטטוסי עסקאות. לצורך אבטחת מידע, הארנק עושה שימוש בהצפנת מפתחות פרטיים, אחסון נתונים מאובטח ב- Firebase.

קישור Vercel: [https://eth-erc-20-wallet-manager.vercel.app](https://eth-erc-20-wallet-manager.vercel.app/)

כניסה אפשרית: שם משתמש: admin, סיסמה: Admin123$

או יצירת חשבון חדש😊

**פונקציונליות מרכזיות לפי סוגי משתמשים**

* **משתמשים חדשים :**

יצירת ארנק חדש עם מפתח פרטי וביטוי שחזור (Mnemonic Phrase)   
 שמירת פרטי הארנק ב-Firebase Firestore לצורך שחזור עתידי  
 אפשרות לשמור נתונים מקומיים ב - LocalStorage

* **משתמשים קיימים :**

שחזור ארנק באמצעות 12 מילים לשחזור  
גישה לנתוני הארנק שלהם, כולל יתרה נוכחית וכתובת הארנק

* **ניהול נכסים :**

הצגת יתרת הארנק ב ETH -ובטוקנים נתמכים  
 עדכון מחירי מטבעות בזמן אמת מ - Binance -וCoinGecko -  
 תמיכה בניהול מספר נכסים

* **שליחת וקבלת נכסים :**

שליחת מטבעות ETH וטוקנים ERC-20 לכתובת אחרת  
 יצירת QR Code לכתובת הארנק לקבלת נכסים.  
 חישוב אוטומטי של מחירי גז (Gas Fees) לפני שליחת עסקה

* **היסטוריית עסקאות :**

שליפת עסקאות מהבלוקצ'יין ישירות מ- Etherscan API  
 תצוגת סטטוס עסקה, סינון עסקאות לפי סוג מטבע ותאריך.

המערכת בנויה לניהול יעיל ובטוח של ארנקי Ethereum עם חוויית משתמש אינטואיטיבית ופשוטה**.**

|  |  |  |  |
| --- | --- | --- | --- |
| |  | | --- | |  |   **רכיב** | |  | | --- | |  |   **טכנולוגיות בשימוש** |
| עיצוב | Tailwind CSS – מסגרת עיצוב מודרנית ל-UI רספונסיבי |
| חזית | React.js & Next.js – פיתוח דינמי מבוסס רכיבים |
| צד שרת | Next.js API Routes – ניהול API עם תמיכה בצד שרת |
| מסד נתונים | Firebase Firestore – בסיס נתונים NoSQL |
| |  | | --- | |  |   אינטגרציה עם בלוקצ'יין | Ethers.js אינטראקציה עם חוזים חכמים ורשת Ethereum |
| |  | | --- | |  |   שליפת מחירים בזמן אמת | Binance API, CoinGecko API – נתוני מטבעות קריפטו מעודכנים |
| ניהול עסקאות | Etherscan API – שליפת מידע על עסקאות בבלוקצ'יין |
| אימות וניהול משתמשים | Firebase Authentication – הזדהות מאובטחת באמצעות Firebase |

**קישור לתיקיית גיט ציבורי:**

<https://github.com/wadeaT/ETH-ERC20-wallet-manager.git>

**דרישות פונקציונליות ולא פונקציונליות**

**דרישות פונקציונליות:**

1. יצירת ארנק Ethereum חדש עם מפתח פרטי וביטוי שחזור.
2. שחזור ארנק קיים באמצעות ביטוי שחזור של 12 מילים.
3. אימות משתמשים.
4. שליחת נכסים.
5. קבלת נכסים.
6. מעקב אחר יתרות הארנק בזמן אמת.
7. שליפת היסטוריית עסקאות ישירות מהבלוקצ'יין.
8. הצגת מחירי מטבעות בזמן אמת.
9. חישוב עמלות גז (Gas Fees) לפני שליחת עסקה.

**דרישות לא פונקציונליות:**

השתמשנו ב wikipedia NFR לסווג דרישות לא פונקצינליות:

**דרישות אבטחה:**

1. אימות משתמשים – כל משתמש חייב להתחבר באמצעות Firebase Authentication.
2. הצפנת נתונים רגישים – אחסון מפתחות פרטיים וביטויי שחזור בצורה מוצפנת.
3. הגנה על עסקאות – שימוש ב- Ethers.js לאימות כתובת מקבל ולמניעת טעויות.
4. שמירה על פרטיות – לא נאגרים נתונים רגישים בשרתים חיצוניים ללא הצפנה.

**דרישות ביצועים:**

1. טעינת מחירים בזמן אמת – שימוש ב- WebSockets לעדכון מחירי מטבעות תוך פחות מ-1 שנייה.
2. ביצוע עסקאות במהירות – העסקאות נשלחות לרשת Ethereum ומאושרות תוך 15 שניות (תלוי בגז).

**דרישות שימושיות:**

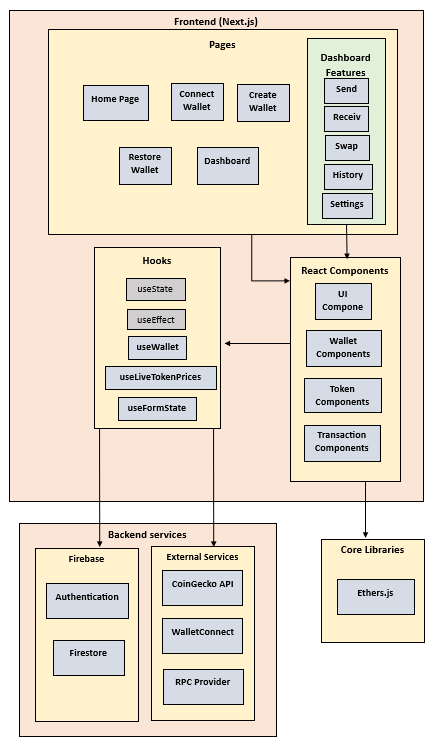
1. ממשק רספונסיבי ואינטואיטיבי – מבוסס על React + Tailwind CSS.
2. עיצוב קל להבנה – שימוש באייקונים ברורים, התראות על שגיאות, וניווט פשוט.

**דרישות תחזוקה:**

1. מודולריות הקוד – שימוש ברכיבי React מודולריים לממשק נקי וקל להרחבה.
2. ניהול גרסאות – הפרויקט משתמש ב- GitHub לניהול גרסאות ועדכונים.
3. אינטגרציה עם Firebase – עדכון קל של מסד הנתונים ללא שינויים בקוד הליבה.

**דרישות סקלאביליות:**

1. תמיכה באלפי משתמשים במקביל – שימוש ב- Firebase Firestore לטעינת נתונים יעילה.
2. אופטימיזציה של קריאות API – שימוש במטמון (Cache) להפחתת קריאות ל- API חיצוניים.

****

**ארכיטקטורה:**

**הסבר לדיאגרמת הארכיטקטורה**

הדיאגרמה מתארת את מבנה הארכיטקטורה של האתר ואת האינטראקציה בין הרכיבים השונים.

**שכבות הארכיטקטורה:**

1. **(שכבת הלקוח**
   * המשתמשים מתחברים דרך ממשק ה- Front-End, שמפותח ב Next.js ומשתמש ב- Tailwind CSS לעיצוב.
   * המשתמש שולח בקשות UI (כגון התחברות, שליחת טרנזקציות, הצגת יתרה) שמועברות לשכבת השרת.
2. **שכבת השרת**
   * בקשות API מתקבלות על ידי Back-End המבוסס על Node.js
   * השרת מבצע עיבוד נתונים, ניהול משתמשים, חישובי יתרות ושליחת עסקאות לרשת הבלוקצ'יין.
   * ה-Back-End מתקשר גם עם מסד הנתונים ומקורות מידע חיצוניים.
3. **מסד נתונים**
   * נתוני המשתמשים, כתובות הארנקים, טרנזקציות ואחסון מידע מוצפנים ונשמרים במסד הנתונים של Firebase
   * השרת שולף נתונים ממסד הנתונים ומעדכן אותם בהתאם לפעולות המשתמש.
4. **ממשקי צד שלישי**
   * האתר משתמש ב- CoinGecko API לקבלת מחירי מטבעות מעודכנים.
   * השרת מבצע קריאות ל- Ethereum RPC API לצורך שליחת טרנזקציות והצגת מידע מהרשת.
   * ה-Front-End יכול גם לשלוח בקשות ישירות לאותם ה-APIs לקבלת מידע בזמן אמת.
5. **פריסת המערכת** 
   * ה- Front-End וה-Back-End נפרסים על גבי Vercel, המאפשר זמני טעינה מהירים ושירות מאובטח בענן.
   * ה-Back-End יכול להיות Node.js ב-Vercel או להתממשק עם שירותי צד שלישי (APIs).

**זרימת הנתונים במערכת:**

1. המשתמש מבצע פעולה בממשק ה- Front-End כגון התחברות, צפייה ביתרה או שליחת טרנזקציה
2. ה- Front-End שולח בקשת API ל- Back-End , שם מתבצע עיבוד המידע.
3. ה- Back-End ניגש למסד הנתונים (Firebase) לקבלת נתוני המשתמש והארנק.
4. אם יש צורך, ה- Back-End מבצע קריאות API חיצוניות לקבלת מידע נוסף (מחירי מטבעות, רשת הבלוקצ'יין).
5. הנתונים מוחזרים ל-Front-End והמשתמש רואה את המידע המעודכן.
6. המערכת מתעדכנת בזמן אמת דרך WebSockets או קריאות חוזרות ל- API

**Use Case:**

הדיאגרמה מציגה את המשתמש ואת הפעולות שהוא יכול לבצע באפליקציית ETH & ERC20 Wallet Manager.

**מקרי השימוש העיקריים:**

1. **יצ**ירת ארנק חדש (Create New Wallet) – המשתמש יכול ליצור ארנק חדש לשימוש באפליקציה.
2. שחזור ארנק (Restore Wallet) – המשתמש יכול לשחזר ארנק קיים באמצעות ביטוי שחזור (Recovery Phrase).
3. חיבור ארנק קיים (Connect Existing Wallet) – המשתמש יכול להתחבר לארנק שכבר יש לו.
4. סקירת תיק ההשקעות (View Portfolio) – מאפשר צפייה בפרטי הארנק, יתרות ומידע נוסף.
5. שליחת טוקנים (Send Tokens - ETH & ERC20) – המשתמש יכול לבצע שליחת מטבעות ETH או ERC20.
6. קבלת טוקנים (Receive Tokens - ETH & ERC20) – המשתמש יכול לקבל מטבעות ETH או ERC20 לארנק שלו.
7. סקירת היסטוריית עסקאות (View Transaction History) – מציג את כל העסקאות שבוצעו בארנק.
8. צפייה בפרטי עסקה (View Transaction Details) – מספק פירוט נוסף על כל עסקה, קשור למקרה השימוש של היסטוריית עסקאות (<<include>> מציין תלות בין שני המקרים).
9. מעקב אחר מחירי טוקנים (Track Token Prices) – מציג למשתמש את השווי של המטבעות בזמן אמת.

תצורה:

* המשתמש (User) מחובר לכל הפעולות שהוא יכול לבצע.
* A diagram of a wallet manager

  Description automatically generatedמקרה השימוש של 'View Transaction Details' תלוי במקרי השימוש של 'View Transaction History', כלומר כדי לצפות בפרטי עסקה יש צורך לגשת להיסטוריית העסקאות תחילה.

**דיאגרמת מבנה DB:**

דיאגרמת מבנה DB:

מסד הנתונים בנוי על Firestore ומנהל מידע על משתמשים וארנקי הקריפטו שלהם. כל משתמש מיוצג על ידי מסמך שבו שם המשתמש בתוך המסמך מאוחסן אובייקט wallet, הכולל את כתובת הארנק ואת המפתח הפרטי (המוצפן) של המשתמש.

מבנה הנתונים:

• כל משתמש מכיל: שם משתמש, תאריך יצירה, תאריך התחברות אחרון.

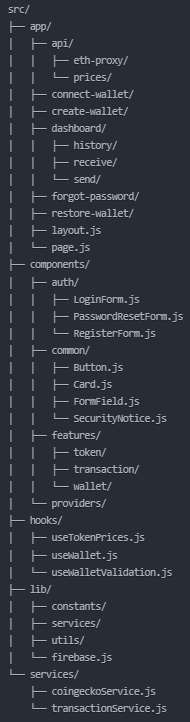
A diagram of a code

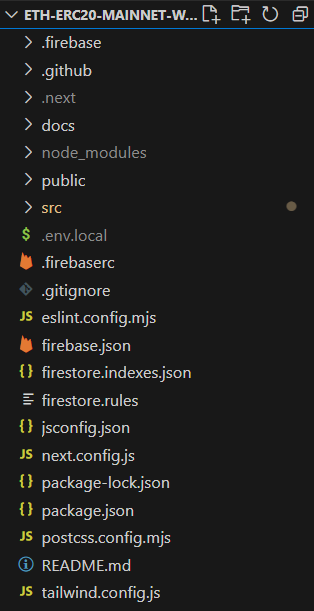
Description automatically generated• כתובת הארנק משמשת למעקב אחר עסקאות.

• המפתח הפרטי נשמר מוצפן לצורך אבטחת מידע.

**מבנה סופי של האתר:**

בתמונה רואים את מבנה התיקיות של הפרויקט בסביבת VS Code:

****

****

**תיק למתכנת:**

סקירת פרויקט

ETH Wallet Manager הוא יישום אינטרנט לניהול ארנקי Ethereum, המאפשר למשתמשים ליצור, להתחבר ולקיים אינטראקציה עם הארנקים שלהם בצורה מאובטחת. האפליקציה תומכת במחירי אסימונים בזמן אמת, היסטוריית עסקאות ופונקציונליות שחזור ארנק.

**טכנולוגיות בשימוש:**

* **Frontend:** Next.js, React, Tailwind CSS
* **Backend:** Firebase Firestore, Next.js API routes
* **Blockchain Interaction:** Ethers.js, Web3 APIs
* **External APIs:** CoinGecko, Binance API, Etherscan API
* **State Management:** React Hooks
* **Database:** Firebase Firestore
* **Authentication:** Firebase Auth

**תכונות עיקריות:**

* Wallet Creation & Restoration
* Secure Authentication & Password Validation
* Live Token Price Updates via WebSockets & API Polling
* Transaction History & Tracking
* Ethereum Mainnet & Token Balances Management
* Real-time Transaction Processing

**הפעלת הפרויקט:**

1. Clone the repository:

git clone https://github.com/wadeaT/ETH-ERC20-wallet-manager.git

1. Install dependencies:

npm install

1. Set up environment variables: Create a .env.local file with the following variables:

NEXT\_PUBLIC\_ETHERSCAN\_API\_KEY=your\_etherscan\_api\_key

NEXT\_PUBLIC\_MAINNET\_RPC\_URL=your\_ethereum\_node\_url

NEXT\_PUBLIC\_MORALIS\_API\_KEY=your\_moralis\_key

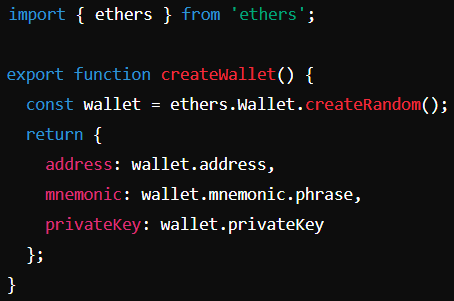
1. Run the development server:

npm run dev

**A screenshot of a computer program

Description automatically generatedתיקיות חשובות עם הסבר:**

**פונקציות מרכזיות:**

**createWallet:**

מה זה עושה?

* יוצר ארנק אקראי (כתובת Ethereum + מפתח פרטי).
* מספק ביטוי זכרון בן 12 מילים לשחזור ארנק.
* מחזירה את כתובת הארנק, המנמונית והמפתח הפרטי.

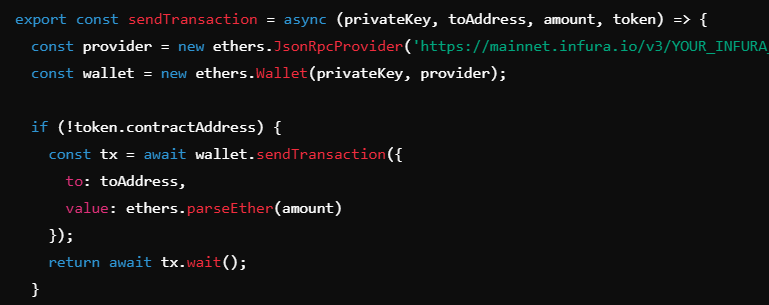
**A screen shot of a computer code

Description automatically generatedRestoring a Wallet from a Mnemonic:**

מה זה עושה?

* מאמת את הביטוי האמנמוני (חייב להיות 12 מילים).
* משחזר את הארנק (מחזיר מפתח פרטי + כתובת).
* עובר לשלב הבא בממשק המשתמש להזנת אישורי משתמש.

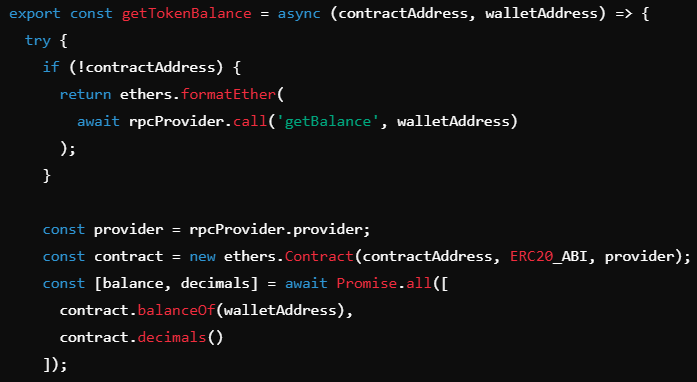
**sendTransactions:**



מה זה עושה?

* אם שולחים ETH, זה קורא sendTransaction().
* אם שולחים אסימוני ERC-20, זה מקיים אינטראקציה עם החוזה החכם.
* משתמש ב-Infura כדי לתקשר עם רשת Ethereum.
* ממתין לאישור העסקה.

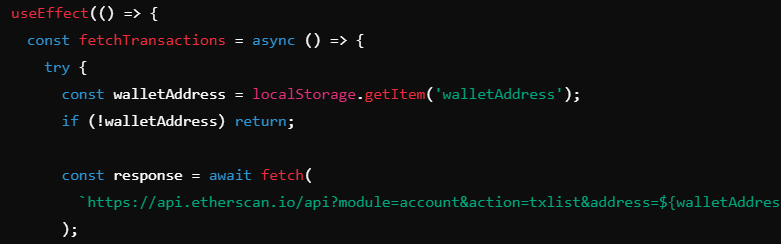
**getTokenBalance:**



מה זה עושה?

* שואב יתרת ETH אם לא מסופקת כתובת חוזה.
* אם זה אסימון ERC-20, הוא שואל את החוזה החכם לאיזון.
* מחזירה את היתרה המעוצבת.

**Transactions History:**



מה זה עושה?

* שואב היסטוריית עסקאות מ-Etherscan.
* מאחסן את העסקאות במצב.
* מציג עסקאות נכנסות ויוצאות

**תקציר:** פונקציות אלה בונות את אפליקציית הליבה

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| | **Where used?** | | --- |  |  | | --- | |  | | |  | | --- | | **What It does?** |  |  | | --- | |  | | |  | | --- | | **Function** |  |  | | --- | |  | |
| Create Wallet Page   |  | | --- | |  | | |  | | --- | |  |   Generates new ETH wallet | |  | | --- | | createWallet() |  |  | | --- | |  | |
| |  | | --- | | Restore Wallet Page |  |  | | --- | |  | | |  | | --- | | Restores a wallet using mnemonic |  |  | | --- | |  | | |  | | --- | | handleRestore() |  |  | | --- | |  | |
| |  | | --- | | Send Transaction Page |  |  | | --- | |  | | |  | | --- | | Sends ETH or tokens |  |  | | --- | |  | | |  | | --- | | sendTransaction() |  |  | | --- | |  | |
| |  | | --- | | Dashboard Page |  |  | | --- | |  | | |  | | --- | | Fetches ETH/token balance |  |  | | --- | |  | | |  | | --- | | getTokenBalance() |  |  | | --- | |  | |
| Transaction History Page | |  | | --- | | Fetches transaction history |  |  | | --- | |  | | |  | | --- | | fetchTransactions() |  |  | | --- | |  | |

**ממשקי API ושירותים חיצוניים:**

1. CoinGecko API (Used in api.js & route.js):

* שואב מחיר אסימון בזמן אמת ונתוני שוק.
* דוגמה ל-API:

fetch(`https://api.coingecko.com/api/v3/simple/price?ids=ethereum&vs\_currencies=usd`)

1. Binance API (Used in priceWebSocket.js & tokenPrices.js):

* stream WebSocket לעדכוני מחירים בזמן אמת.
* דוגמה ל-:API

const ws = new WebSocket(`wss://stream.binance.com:9443/ws/ethusdt@ticker`);

**חיבור למסד נתונים:**

Firebase Firestore:

קונפיגורציה:

const firebaseConfig = {

apiKey: process.env.FIREBASE\_API\_KEY,

authDomain: "eth-wallet-xxxxx.firebaseapp.com",

projectId: "eth-wallet-xxxxx",

};

**קישור לFirebase :**

https://console.firebase.google.com/project/eth-wallet-62568/overview

**תיק משתמש:**

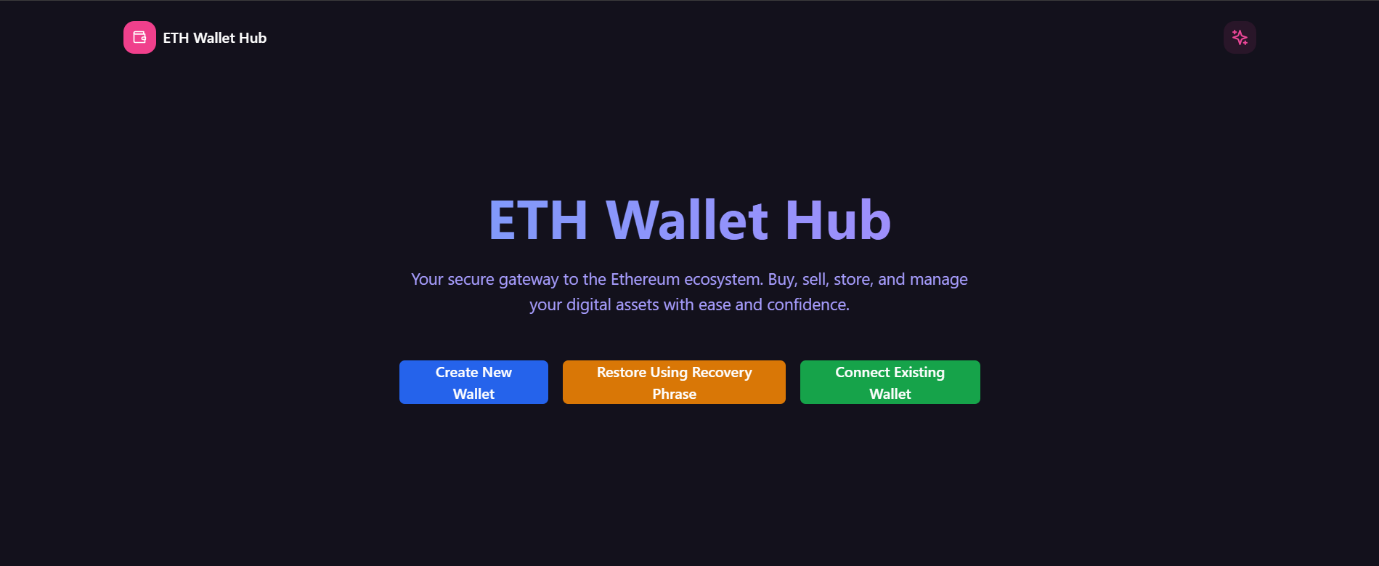
ברוכים הבאים למדריך המשתמש של ETH & ERC20 Wallet Manager – אפליקציה לניהול ארנקי Ethereum ו-ERC20 בצורה מאובטחת, מהירה ונוחה.

אפליקציה זו מאפשרת למשתמשים ליצור ארנק חדש, לשחזר ארנק קיים, לשלוח ולקבל מטבעות, לעקוב אחר מחירי טוקנים ולצפות בהיסטוריית עסקאות – הכול בפלטפורמה אחת אינטואיטיבית.

**למי האפליקציה מיועדת?**

**משתמשים חדשים** שרוצים להתחיל להשתמש בארנק דיגיטלי.  
**משתמשים קיימים** שרוצים לנהל את הארנק שלהם בקלות.  
**משקיעי קריפטו** שזקוקים לכלי לניהול ושליחת טוקנים באופן יומיומי.

מכאן נמשיך להסבר מפורט על כל אחד ממסכי האפליקציה וכיצד להשתמש בהם

1. **מסך ראשי:**

מסך הכניסה הראשי של האפליקציה מציע שלוש אפשרויות לבחירת המשתמש:

1. **יצירת ארנק חדש** – מיועד למשתמשים חדשים שמעוניינים להתחיל להשתמש בארנק.
2. **שחזור ארנק** – למשתמשים שכבר יש להם ארנק ומילות שחזור.
3. **חיבור ארנק קיים** – מיועד למי שיש כבר ארנק פעיל ורוצה להתחבר אליו.

**אפשרויות נוספות:**

* **כפתור שינוי מצב יום/לילה** – ממוקם בפינה הימנית העליונה ומאפשר מעבר בין מצב תאורה כהה לבהירה בהתאם להעדפת המשתמש.

1. A screenshot of a computer

   Description automatically generatedמסך יצירת ארנק חדש:

במסך זה תוכלו ליצור חשבון חדש על ידי הזנת הפרטים הבאים:

1. **שם משתמש** – בין 3 ל-20 תווים.
2. **סיסמה** – חייבת להכיל לפחות 8 תווים, כולל אותיות, מספרים ותווים מיוחדים.
3. **אימות סיסמה** – יש להקליד את הסיסמה שוב לאימות.

**תכונות נוספות:**

* בצד ימין מוצגות הדרישות ליצירת חשבון חדש.
* לאחר מילוי כל השדות, יש ללחוץ על כפתור **"Continue"** כדי להמשיך לשלב הבא

1. A screenshot of a computer

   Description automatically generatedמסך שחזור ארנק:

במסך זה ניתן לשחזר את הארנק הקיים באמצעות **12 מילות השחזור**:

1. הזינו את **כל 12 מילות השחזור** בסדר המדויק שלהן, מופרדות ברווחים.
2. ניתן להדביק את כל הביטויים באמצעות כפתור **"Paste Phrase"**.
3. לאחר הזנת הביטויים, לחצו על **"Restore Wallet"** כדי להשלים את השחזור.
4. מסך התחברות:

A screenshot of a login form

Description automatically generated

מסך זה מאפשר כניסה לארנק קיים באמצעות:

1. **הזנת שם משתמש**.
2. **הזנת סיסמה** (ניתן להציג את הסיסמה על ידי לחיצה על סמל העין).
3. לאחר הזנת הנתונים, יש ללחוץ על כפתור **"Connect Wallet"** כדי להתחבר.
4. A screenshot of a computer

   Description automatically generated**מסך סקירה כללית של הארנק:**

תפריט צד שמאל:

* Overview - סקירה כללית
* Send - שליחת כספים
* Receive - קבלת כספים
* Swap - המרת מטבעות
* History - היסטוריית פעולות
* Settings - הגדרות
* Theme - שינוי ערכת נושא
* Logout - התנתקות

בחלק העליון מוצג:

* כתובת הארנק (ניתן להעתיק)
* שווי כולל
* מספר הטוקנים הפעילים

בגוף המסך: רשימת המטבעות עם היתרה והשווי של כל אחד:

1. **מסך שליחת מטבעות**

A screenshot of a computer

Description automatically generated

המסך מאפשר לשלוח מטבעות קריפטו:

1. **בחירת מטבע** - בחרו את המטבע לשליחה
2. **כתובת מקבל** - הכניסו את כתובת הארנק של המקבל
3. **סכום** - הזינו את הכמות לשליחה
   * ניתן להשתמש בכפתור Max לשליחת כל היתרה
   * מוצג שווי ב-USD ויתרה נוכחית

לסיום לחצו על כפתור "Send" לביצוע ההעברה.

1. מסך קבלת מטבעות:

A screenshot of a computer

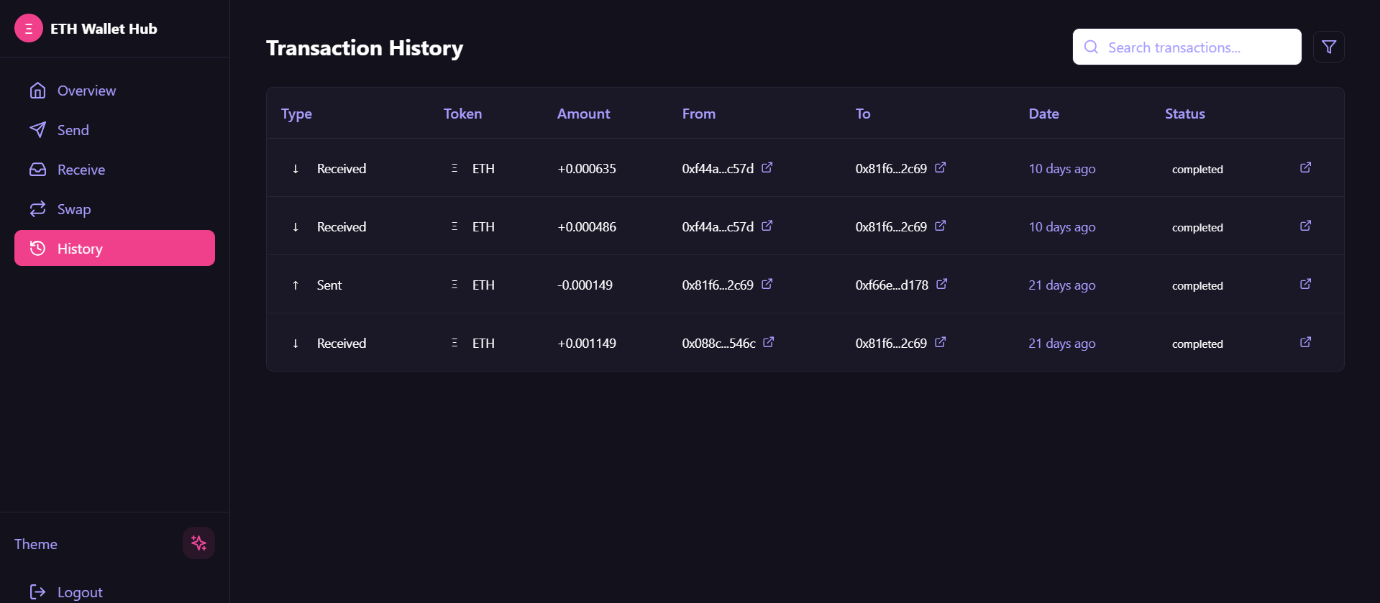
Description automatically generated

במסך זה תמצאו:

1. **קוד QR** - לסריקה מהירה של כתובת הארנק
2. **כתובת הארנק** שלכם - ניתן להעתיק בלחיצה על סמל ההעתקה

**שימו לב**: ניתן לשלוח לכתובת זו רק ETH ומטבעות מסוג ERC-20. שליחת מטבעות אחרים עלולה לגרום לאובדן!

1. מסך היסטוריית עסקאות:



במסך זה תוכלו לצפות בכל העסקאות שבוצעו באמצעות הארנק. עבור כל עסקה מוצגים הפרטים הבאים:

* **סוג הפעולה** (שליחה - Sent / קבלה - Received).
* **סוג המטבע**.
* **כמות** ( + לקיבול, - לשליחה).
* **כתובת השולח**.
* **כתובת המקבל**.
* **תאריך ושעה שבה בוצעה הפעולה**.
* **סטטוס העסקה** (הושלם / בהמתנה / נכשל).

**אפשרויות נוספות במסך:**

* **שורת חיפוש** – לאיתור עסקאות מסוימות.
* **כפתור סינון** – מאפשר סינון העסקאות לפי תאריך, סוג מטבע או סוג פעולה.