Micro-Task and Earning Platform

Hello Candidate,

Welcome to the job assessment for the position of Junior MERN Stack Developer. This assessment consists of a project that you will need to complete using the MERN stack. This project is designed to evaluate your MERN-Stack technologies and skills. **You need to complete this project using Vibe coding. You can use any tools (Github Copilot, cursor, antigravity, windsurf, Trae, Figma make, Bolt, Lovable, V0, etc.) as you like. This assignment is totally optional, but it's the best way to judge your learning. So don't skip this.**

Key Rules:

- Include a minimum of 20 notable GitHub commits on the client side.
- Include a minimum of 12 notable GitHub commits on the server side
- Add a meaningful README.md file with the name of your website, Admin username, password, and live site URL. Include a minimum of 10 bullet points to feature your website.
- Make it responsive for all devices. You need to make it responsive for mobile, tablet, and desktop views. Make the dashboard responsive as well.
- After reloading the page of a private route, the user should not be redirected to the login page.
- Use the Environment variable to hide the Firebase config keys and MongoDB credentials.
- Don't use any Lorem ipsum text on your website.

Tips for You

- Read and understand the assessment theme carefully.
- Select a design and stay stuck with the same color and layout.
- Don’t go through the whole requirement at once. Complete the requirements one by one.

Good Luck! We wish you the best of luck and look forward to seeing your technical prowess and innovative solutions

---

Overview

The Micro Tasking and Earning Platform is designed to provide users with opportunities to complete small tasks and earn money.
You can take design inspiration from micro-earning sites like Picoworkers. Clickworker, SEOClerks.But don't copy the design.
The platform accommodates 3 distinct roles: **Worker**, **Buyer**, and **Admin**. Each role is tailored with specific functionalities to ensure seamless task management, task creation, and platform administration.

1. **Worker**: Completes tasks to earn rewards by viewing tasks, submitting them for review, withdrawing coins, and receiving notifications.
2. **Buyer**: Manages tasks and payments by creating tasks, reviewing submissions, paying Workers, purchasing coins, and reporting issues.
3. **Admin**: Oversees platform operations by managing user roles, addressing reports, and maintaining system integrity.

Main Requirements

1. Layout Structure

The layout for the Micro Tasking and Earning Platform will be divided into two primary structures: **Basic Layout** and **Dashboard Layout**. We will talk about the Dashboard layout after 4th Requirement

Navbar will serve as the primary navigation tool for users, offering easy access to the platform's main features and functionalities. Navbar will contain the following Navigations

For Not Logged in Users

- Website Name / Logo
- Login
- Register
- Join as Developer (This Button will redirect you to your Client Github Repository.)

For Logged Users

- Website Name / Logo (By clicking, it will redirect the user to the home page)
- Dashboard
- Available Coin
- User Profile and Logout Button
- Join as Developer (This Button will redirect you to your Client Github Repository.)

Footer will contain the website Logo, and linkable social media icons that will redirect users to your profile, such as LinkedIn, Facebook, GitHub, etc.

2. Home Page

The Homepage will be the first impression of the platform, so it needs to be engaging, informative, and visually appealing. You must use Animation on the Homepage.

- **Hero Section**

  It will contain a slider with three banners. Use React-Responsive Carousel / Swiper Slider. Each slide will contain a different heading and Title. You can also use a background video instead of a slider.

- **Best Workers**

  Show the Top 6 Workers' data who have the maximum coins. Show their picture and available coins.

- **Testimonial Section:**

  Displays feedback from satisfied users in a slider format. Includes user photos, names, and brief quotes about their positive experiences. **This section will be static**. Use the swiper slider to build this section.

- **3 Extra Sections:** Create a minimum of 3 Extra sections with your own idea.

---

3. User Authentication System

The user authentication system will manage the registration and login processes for the platform. It ensures that users can securely access the platform and that the appropriate roles and permissions are assigned. Here's a detailed description of the system:

1. Registration Page

Here, Users can create an account by providing the necessary information. **After registration**, the **worker will get 10 coins,** and the **Buyer will get 50 coins** by default.

There will be 2 Types of Registration methods that have to be implemented here.

A) Form with Input Fields:

- Name + Email + Profile Picture URL + Password
- Drop-down to select the role
    - Worker
    - Buyer

*** Implement Input validation for email format and password strength. Show error messages for invalid input, such as an existing email.

After registration, the **worker will get 10 coins,** and the **Buyer will get 50 coins** by default.
ensure that users will only get the coin on registration.

> 💡You need to store user info with coin value in the database

2. Login Page

Here, Users can sign in by providing the necessary information. There will be 2 Types of Login methods that have to be implemented here.

- Users can log in using their registered email and password.
- Google Sign-In option for quick authentication.

*** Implement Input validation for incorrect email and password. After successful Login/registration, redirect the user to the Dashboard.

*** After Login / Registration you have to store a secret access-token for users in their browser local storage.

DASHBOARD

Layout

Dashboard will be Look Like This design

Logo	Available coin | userImage userRole | userName	Notification
Navigation	Sections Based on Routes	
	Footer	

Dashboard will Show Following Navigation based on user Role.

Worker	**Buyer**	**Admin**
Home	Home	Home
TaskList	Add new Tasks	Manage Users
My Submissions	My Task’s	Manage Task
withdrawals	Purchase Coin	
	Payment history	

Dashboard for Buyer

1. Buyer-Home

States

Buyer will see his total task Count (task added by user), pending Task(sum of all required_workers count of his added Tasks), and total payment paid by the user.

Task To Review

Buyer will see all submissions for his added tasks where the status is “pending” in a table format with the following information

- worker_name
- task_title
- payable_amount
- View Submission Button(will open a modal and show the submission detail)
- Actionable Buttons
    - Approve Button
    - Reject Button

*** On Clicking the Approve Button

- increase payable amount coin for the workers
- Change the SubmissionStatus to “approve” for specific submissions.

*** On clicking the Reject Button,

- change the status to “rejected” of the specific submission.
- Increase required_workers by 1.

2. Add new Tasks

This section will contain a Form with the following input fields

- task_title (ex: watch my YouTube video and make a comment)
- task_detail // detail description
- required_workers: (number) , // Total number of workers needed ex. 100
- payable_amount (number) , // Amount will pay to each worker ex. 10
- completion_date // Deadline for completing the Task
- submission_info // what to submit, like screenshot / proof
- task_image_url // image to attract workers

  (Implement imageBB for uploading if you want to get a challenge mark)
- **Add Task Button**. On clicking Add Task, you have to do the following
    - Calculate **Total payable amount** (required_workers * payable_amount)
    - If **Total payable amount** is greater than user's available coin, then

        - through an alert “Not available Coin. Purchase Coin ”,
        - terminate the process &
        - Navigate users to the Purchase Coin Page.
    - Else, Do following
        - Save the Task into a collection
        - Reduce Buyer’s coin.

3. My Task’s

In this section, the user will show all the tasks he added in descending order based on compilation date in a table format.

- Show The task info in columns with an update and delete button
- onClicking update, users can update the **Title**, **TaskDetail,** and **submission_Details**.
- onClicking Delete,
    - delete the task from the task Collection
    - Calculate refill amount (required_workers * payable_amount)
    - Increase the coin for unCompleted tasks in his available coin.

4. Purchase Coin

From This Route, the User can purchase coins. Implement a stripe-based payment system on this route.

Payment info

Show users 4 cards like the following info. This is a demo. You can change the quantity design with your ideas.

10 coins =	**150 coins** =	**500 coin** = **20$**	**1000 coin** = **35$**

1 $	**10$**		

Clicking a specific card redirects the user to pay a specific amount. Implement a stripe-based **payment system** on it. I know you can do it. After successful payment

- Save the payment info
- Increase bthe uyer's coin

If you cant integrate stripe payment add a dummy payment and move on

5. Payment History

Show All the payments made by the Buyer in the payment history route in a tabular format.

Dashboard For Worker

1. Home Page (For Worker)

States

Workers will see the Total Submission (Count of all submissions made by the worker), Total pending submission (Count of all submissions made by the worker where status is pending), and Total Earning (sum of payable_amount of the worker where status is approved).

Approved Submission

Worker will see all the submissions made by him where the status is "approved " in a table format from the submission collection, with the following information

- task_title
- payable_amount

- Buyer_name
- status

2. TaskList

In This Route, the Worker will see All the tasks where the required_worker is greater than 0, with the Following information

- task_title
- Buyer_name
- completion_date
- payable_amount
- required_workers
- View Details Button

Data will be in card format. By clicking View Details, navigate workers to the task details route.

3. Task Details

show all the information of the Task and a submission form in this Route.
The Submission form will contain 1 input field(text-area) name submission_Details.
After submitting the form, insert, save the submission in the database with task_id, task_title, payable_amount, worker_email, submission_details, worker_name, Buyer_name, Buyer_email, current_date, and a status (pending).

4. My Submission

Show all the submissions where the workerEmail matched the login worker Email.
show data in a tabular form (which data you want to show is upto you). Highlight the submission status.

5. WithDrawals

Note: remember buyer is purchasing 10 coins for 1 dollar. But the worker will withdraw 1

dollar with 20 coins. That's the business logic of how the platform is earning 😎

20 Coins = 1 Dollar.

Users can withdraw when they have a minimum of 200 coins, which is equivalent to 10 dollars.

User Total Earning

Show the user their current coin and withdrawal amount in dollars.

For example, if a worker has 300 coins, then his withdrawal amount is 15 dollars.

WithDrawal Form

- Coin To Withdraw (Number) // It can not exceed the total coin.
- Withdraw_amount ($) (Number) (Not editable . It will change when the coin to withdraw field changes. 20 coins = 1 dollar)
- Select Payment System (DropDown) (Stripe (you need to implement), Bkash, Rocket, Nagad, or others you can add (these are optional, you can just add for design purpose or if you want to experiment, you can try))
- Account Number
- Withdraw Button. If the user doesn't have enough coin make this button disappear and show a text "Insufficient coin".

OnClick withdrawal button saves the data into the database with worker_email, worker_name, withdrawal_coin, withdrawal_amount, payment_system, withdraw_date, and a status (pending)

Dashboard for Admin

1. Admin-Home

States

Admin will see the count of total worker, total buyer, total available coin(sum of all users coin), and total payments

Withdraw request

Admin will see all withdrawal requests from withdrawCollection made by users where the status is pending in a table format with a payment success button.

After clicking the payment success button,

- Change the withdrawal request status to approved.
- Decrease user coin by the withdrawal amount.

2. Manage Users

The section will show a table of all users with display_name, user_email, photo_url, role , coin and some actionable button

- remove (will delete the user from the database)
    - By clicking Remove user will be deleted from the server.
- Update Role (Dropdown field. On change, it will change the role of the user)
    - Admin
    - Buyer
    - Worker

3. Manage Tasks

Admin will see the TaskList in a table format with task information in a table format with Delete Task (By clicking Task will be deleted from the database) button.

Challenges

1. Secure Authorization

Implement Role-Based Authorization for users. Create middleware for Worker, Admin, and Buyer. (You can use the Firebase Admin SDK for authentication and authorization)

2. Notification System

Implement a notification System. Suppose if a Buyer changes a submission status on submission collection, then add a notification in the notification collection in the following format

Example

```
{
        message: " you have earned {payable_amount} from {BuyerName} for completing {taskTitke}
        ToEmail: {workerEmail}
         actionRoute : {/dashboard/worker-home}
        Time: new Date()
}
```

Do this for **approval / reject the submission by the buyer to the worker**, **approving the withdrawal request by the admin to worker**. **Inserting a new submission by the worker to the buyer**

On clicking the notification icon, show all the notifications where the email of the "toEmail" key matches the current user email, sorted in descending order.

Show notification in a floating pop-up. By clicking anywhere on the page, pop up will be hidden

3. Add Pagination on My Submission Page for Worker

Implement pagination on the My Submission Route.

4. Image Uploading System with imgBB

Implement an image uploading System with imageBB on **registration,** and **Add new Task** Route.

Additional information:

- You can host images anywhere.

- You can use vanilla CSS or any library.
- Try to host your site on Firebase (Netlify hosting will need some extra configurations)
- Make Sure you deploy server-side and client-side on the first day. If you have
- any issues with hosting or GitHub push, please join the "GitHub and deploy" related support session.

What to submit:

Admin email:

Admin password:

Front-end Live Site Link:

Client Side Github Repository Link:

Server Side Github Repository Link:

Optional

You can also add the following features if you implemented them

1. **Automated Email Notifications:**
   - Set up automated email notifications for various actions (e.g., task approval/rejection, payment confirmation, withdrawal processing).
   - Use services like SendGrid or AWS SES for sending emails.

2. Advanced Search and Filter Functionality:

- Implement a comprehensive search and filter system for tasks. Users should be able to filter tasks based on criteria like task type, deadline, reward amount, and status.

- Use MongoDB's aggregation framework for efficient querying and filtering on the server side.

3. Report System for invalid Submission

- Implement a reporting system for invalid submissions. So that the admin can take proper action on the user.