Value Time Assessor

Cohort # WD 112

Table of Contents

Table of Contents

Mission Statement

User Stories
- User 1: The guest

Database
- Tables
- Endpoints

Features

Wireframe

Schedule

Final Notes

Mission Statement

This app is designed to help individuals define a dollar amount they associate with the value of their time, track the amount of time they spend on neutral tasks, and then determine which tasks are most cost-efficient to outsource and which they should complete themselves.

User Stories

User 1: The frugal one

As a money-conscious individual, this user wants a way to determine the cost-benefit ratio of saving money upfront in the short-term versus how much money it really saves them in the long-term

User 2: The quick spender

As someone who has a hard time saving money, this user wants a way to determine which conveniences they can keep paying for and how they could save money by cutting out some easy-spending options that aren't crucial or necessary

User 3: The time-conscious individual

As someone who has a lot of personal tasks and limited time outside of a full work schedule to complete them, this user wants a way to decide which tasks can be outsourced to save time and free up time for more relaxing time or more pressing/important tasks

User 4: The flitter

As someone who struggles to focus on one task at a time, this user wants a way to prioritize their tasks by determining which they most enjoy in conjunction with which need to be completed chronologically

Database

Tables

Table 1: Users

id	*firstName*	*lastName*	*email*	*password*
num	*string*	*string*	*string*	*string*

Table 2: Time Value

id	*hourlyWage*	*neutralValue*	*userId**
num	*num*	*num*	*num*

*Association: userId is id from table 1

Table 3: Time of Tasks

id	*cleaning*	*laundry*	*mealPrep*	*petCare*	*shopping*	*carCare*	*taxes*	*userId**
num	*num*	*num*	*num*	*num*	*num*	*num*	*num*	*num*

*Association: userId is id from table 1

Stretch goal Table 4: Time Value Calculator

id	*cleaningValue*	*laundryValue*	*mealPrepValue*	*petCareValue*	*shoppingValue*	*carCareValue*	*taxesValue*	*userId**	*timeId**	*taskId**
num	*string*	*string*	*string*	*string*	*string*	*string*	*string*	*num*	*num*	*num*

*Associations: userId is id from table 1, timeId is id from table 2, taskId is id from table 3

Endpoints

Users: /auth

- POST /register Registers a new user account
- POST /login Logs in a user
- GET /userinfo Retrieves user information and data

Time Value: /values

- POST /value Users log values for time
- GET /value Users retrieve their values for time
- PUT /value Users update their values for time
- DELETE /value Users delete their values for time

Time of Tasks: /timetasks

- POST /tasks Users log times for tasks using stopwatch
- GET /tasks Users retrieve their time for tasks
- PUT /tasks Users update their times for tasks using stopwatch
- DELETE /tasks Users delete their times for tasks

Time Value Calculator: /timevalue

- GET /timevalue Users retrieve their time to value calculation
- DELETE /timevalue Users delete their time to value calculation

Features

Version 1.0 / MVP	Version 2.0 / Stretch Goals
● Users can create accounts ● Users can log in and log out ● Users can input their hourly wage to calculate the value of their time ● Users can track their time on neutral tasks using a stopwatch built into the app ● The app will save the user's time spent on neutral tasks and average that time over the course of a week or month ● The app will suggest tasks to be paid for based on the user's time-to-value dollar amount ● Users can save their own information to access later ● Users can update their value of time to recalculate tasks to spend time on	● Users can take an assessment to calculate the value of their neutral time ● Users can reset passwords ● Users can find services available based on what the app says would be beneficial to them to pay for ● User can vote up or down the enjoyability they have for neutral tasks

Data Flow

Server:

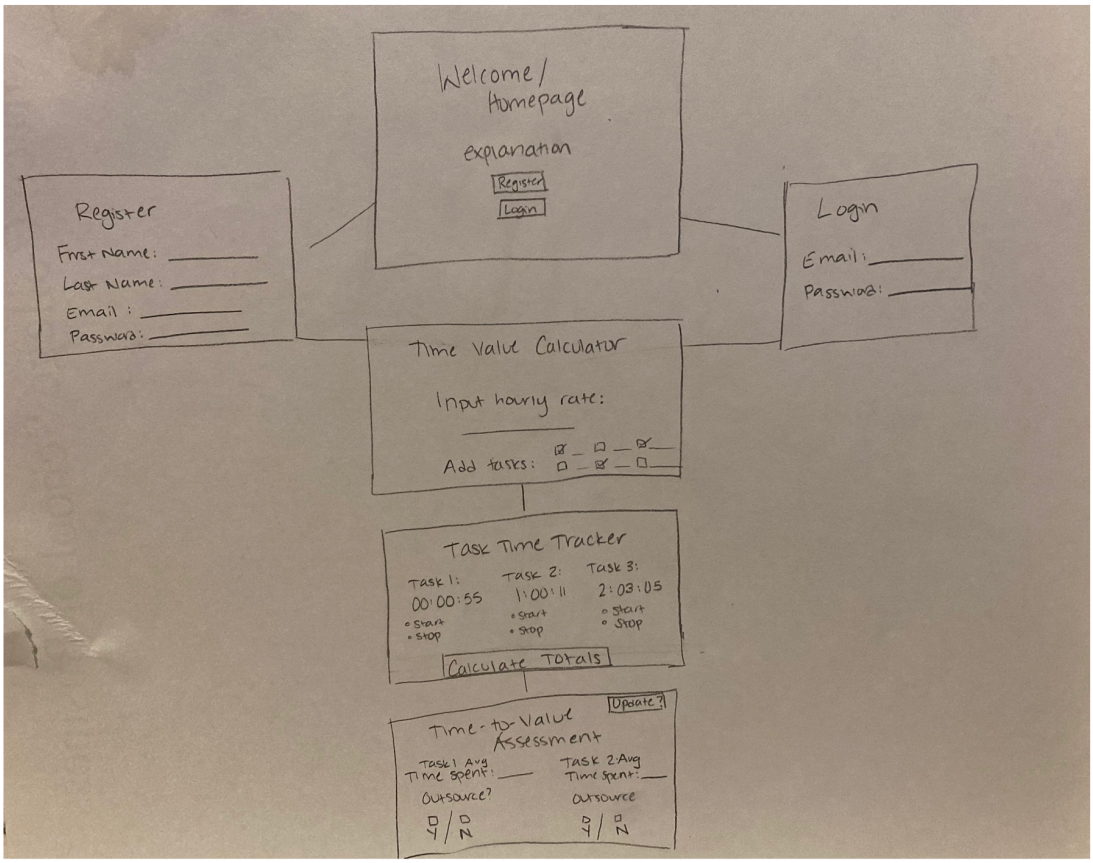
- Controllers:
    - Userscontroller → import models, bcrypt, jwt; use UsersModel specifically
    - Timevaluecontroller → import models; use TimeValueModel specifically
    - Timetaskscontroller → import models; use TimeTaskModel specifically
    - Valuescontroller → import models; use ValuesModel specifically
    - Index → export models
- Middleware:
    - Index → export CORS and validateSession
    - Validate-session → import models and jwt, export validateJWT
    - Headers
- Models:
    - Users → import db, create user id, firstName, lastName, email, password
    - Timevalue → import db, create
    - Timetask → import db, create
    - Values → import db, create
    - Index → import db, import UsersModel, TimeValueModel, TimeTaskModel, ValuesModel, export dbConnection, models
- .Env
- .Gitignore → .env
- App → import dbConnection, controllers, middleware
- Db → dom .env

Client:

- Components:
    - Auth
        - Auth → import register, login
        - Register → props, user
        - Login → props, user
    - TimeValue
        - TimeValueTable → props, token, headers
        - TimeValue → import TimeValueTable
    - TimeOfTasks
        - TimeOfTasksTable → props, token, headers
        - TimeOfTasks → import TimeOfTasksTable
    - Values
        - ValuesTable → props, token, headers

        - Values → import ValuesTable
- App.css
- App.js → import from components
- Index.css
- Index.js → import ReactDOM, index.css, App

Wireframe



Color palette: https://coolors.co/f4f1de-e07a5f-3d405b-81b29a-f2cc8f

Notes to self: need to have certain tasks with average dollar amount of cost of services attached:

- Cleaning/organizing your house (vacuuming, cleaning the kitchen, cleaning the bathroom, picking up toys, organizing files)
- Laundry
- Meal prep
    - Planning a grocery list, grocery shopping (in-store or online), picking up groceries (if shopped online)
    - Preparing meals
    - Doing dishes
- Pet care
    - Taking pet out
    - Grooming (nail trimming, wash, hair brush/trim)

    - Walks/play time
- Shopping (other than for groceries)
    - Clothes
    - Other items
- Car maintenance/care
    - Car wash
    - Filling gas
    - Checking/filling tires
    - Oil change
    - Routine maintenance
    - Detailing
- Taxes

Schedule

Jan 10	Jan 11	Jan 12	Jan 13	Jan 14	Jan 15	Jan 16
Server: create models, middleware, controllers	Server: create models, middleware, controllers	Server: test in pgadmin/ postman	Server: check token validation, role based access control	Server: check database associations, create readme	Server: error handling	Server side complete
Jan 17	Jan 18	Jan 19	Jan 20	Jan 21	Jan 22	Jan 23
Client: create signup and login	Client: create value calculator	Client: create stopwatch function	Client: create value to time display	Client: check validation, react router, lifecycle methods, props/state, patterns	Client: Styling, error handling	Client side complete
Jan 24	Jan 25	Jan 26				
Deploy to Heroku	Trouble-shooting/ final stretch-goal additions	Final Project Complete/ Present-ations				

Final Notes

Great job with planning! You are now set to start coding. Planning a project is incredibly beneficial to your success and the success of your project. Here are some resources to help you with your planning.

- How to plan a web application
- Step By Step: Planning a web application