

Time Detective Services & API Endpoints

Mike Colbert

Table of Contents

HTTP Verb	API Endpoints	Purpose	Page
POST	/user	Create a new user	p. 2
GET	/user/login	Log in to application	p. 3
GET	/user/logout	Log out of application	p. 4
GET	/user/{user_id}	Get details of single user	p. 5
PUT	/user/{user_id}	Update a user's details	p. 6
DELETE	/user/{user_id}	Delete a user	p. 7
GET	/logs/summary	Summary of user's tasks	p. 8
GET	/logs/{task_id}	List of all log entries for a task	p. 9
GET	/log/{log_id}	Return an individual log entry	p. 10
PUT	/log/{log_id}	Update an individual log entry	p. 10
DELETE	/log/{log_id}	Delete an individual log entry	p. 11
GET	/tasks	Return a list of tasks being tracked by the user	p. 12
GET	/task/{task_id}	Return an individual task	p. 13
PUT	/task/{task_id}	Update an individual task	p. 13
DELETE	/task/{task_id}	Delete an individual task	p. 14
POST	/task	Create a new task to be tracked	p. 15
POST	/log/start	Start a timer	p. 17
POST	/log/stop	Stop a timer	p. 17
POST	/log/manual	Manually enter an event in the log	p. 18
GET	/admin/users	List all users of the application	p. 19
DELETE	/admin/user/{user_id}	Delete a user from the application	p. 20
GET	/admin/user/{user_id}	Get an individual user's details	p. 21
PUT	/admin/user/{user_id}	Update an individual user	p. 21
PUT	/user/{user_id}	Update a user's details – pomodoro timer	p. 22
POST	/log/interrupt/{task_id}	User was interrupted during their task	p. 23
POST	/log/resume/{task_id}	User has resumed working on task after interruption	p. 24
POST	/user/reset/{user_id}	Reset user's password	p. 25
PUT	/task/{task_id}	Archive / unarchive a task	p. 26

Create a New Account: MVP

The screenshot shows a web browser window with the URL `http://timedetective.io/createAccount`. The page title is 'Time Detective' and it has a 'Login' link. The main heading is 'Create a New Account'. The form contains the following fields and labels:

- First name:
- Last name:
- * Email:
- * Password:
- * Re-enter Password:

Below the fields is a 'Create Account' button. At the bottom of the browser window, there is a footer that says 'Please Note: I Agree' and 'Time Detective 2021'.

Form data POST to /user

success message: Your account was created successfully. → /dashboard

error message: This email address is already registered.

error message: Passwords do not match.

error message: Required data is missing.

Page	HTTP Verb	API Endpoint	Description	CRUD Operation
Create User /createUser	POST	/user	Create a new application user from form data.	CREATE
<p><i>Parameters:</i></p> <p><i>Body:</i></p> <pre>{ "first_name": "Mike", "last_name": "Colbert", "email": "mike@mike.com", "password": "password_in_plain_text" }</pre>				
<p><i>Response:</i></p> <p>Code: 200 Description: Successful operation</p> <p>Code: 405 Description: Invalid input</p>				

Login: MVP

Time Detective

Login to Time Detective

* email:

* password:

☐ Remember me

[Forgot password](#)

Please Refer Us About Time Detective 2021

Form data GET to /user/login

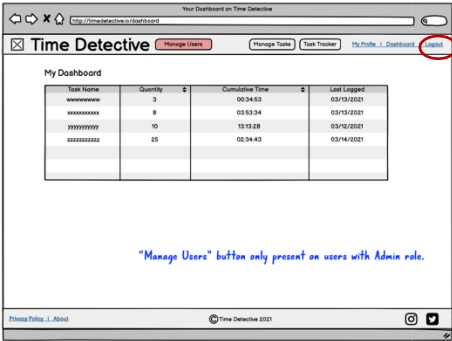
success: redirect to Task Tracker page (/tracker)

error message: Email address not found.

error message: Password error.

Page	HTTP Verb	API Endpoint	Description	CRUD Operation
Login /login	GET	/user/login	Authenticate the user	SELECT
<p>Parameters:</p> <p>Body:</p> <p>Login form:</p> <ul style="list-style-type: none">email - The email address for loginpassword - the clear text password associated with the email address <p>X-Expires-After - string - date in UTC when token expires <i>force user to re-authenticate periodically</i></p> <p>X-Rate-Limit - integer - calls per hour allowed by the user <i>prevent brute force password guessing</i></p>				
<p>Response:</p> <p>Code: 200 Description: Successful operation</p> <p>Code: 400 Description: Invalid username/password supplied</p>				

Logout: MVP



Time Detective

My Dashboard

Task Name	Quantity	Cumulative Time	Last Logged
wwwwww	3	00:34:53	03/13/2021
xxxxxxxxxx	8	03:53:34	03/13/2021
xxxxxxxxxx	10	13:13:28	03/13/2021
xxxxxxxxxx	25	00:34:43	03/14/2021

"Manage Users" button only present on users with Admin role.

Logout from any page

Form data GET to /user/logout

success: redirect to home page (/index)

Page	HTTP Verb	API Endpoint	Description	CRUD Operation
logout /logout	GET	/user/logout	Log out the current user from the system	---
<div>Parameters: No parameters</div>				
<div>Response: Code: 200 Description: Successful operation</div>				

My Profile: MVP

List my profile details

→ GET to `/user/{user_id}`

← User details returned to page

success: page displays user's details

error message: User not found.

Page	HTTP Verb	API Endpoint	Description	CRUD Operation
My Profile /profile	GET	/user/{ user_id }	Get user details from database by user id	SELECT
<div>Parameters:</div> <div>user_id</div>				
<div>Response:</div> <div> Code: 200 Description: Successful operation <pre> { "user_id": 0, "first_name": "user_first_name", "last_name": "user_last_name", "email": "user_email_address", "role": 1, "password": "hashed_password", "date_created": date, "last_login": date } </pre> </div> <div> Code: 400 Description: Invalid user id supplied </div> <div> Code: 404 Description: User not found </div>				

Use userID 1 for testing.
Can only be done by a logged in user.

My Profile: MVP

Update my profile fields

The screenshot shows a web browser window with the URL 'http://timedetective.io/profile'. The page title is 'Time Detective'. The main content area is titled 'My Profile'. It contains several input fields: 'First name', 'Last name', 'Email', 'Change Password', and 'Re-enter Password'. Below these fields are two buttons: 'Update Account' and 'Delete Account'. At the bottom, there are two lines of text: 'Last login' and 'Date created', both followed by a placeholder text 'user_email_address user_email_address'. The footer of the page says 'Please Refer: J. About' and 'Time Detective 2021'.

PUT to /user/{user_id}

Redirected to "My Profile" page

success message: Your account was successfully updated. → /myprofile

error message: This email address is already registered.

error message: Passwords do not match.

error message: Required data is missing.

Page	HTTP Verb	API Endpoint	Description	CRUD Operation
My Profile /profile	PUT	/user/{ user_id }	Update user details in database by user id	UPDATE
Use userID 1 for testing. Can only be done by a logged in user.	<div>Parameters:</div> <div>Body:</div> <pre>{ { "user_id" : 0, "first_name" : "user_first_name", "last_name" : "user_last_name", "email" : "user_email_address", "password" : "plain text_password", "password2" : "plain text_password" } }</pre>			
	<div>Response:</div> <div>Code: 200 Description: Successful operation</div> <div>Code: 400 Description: Invalid user id supplied</div> <div>Code: 404 Description: User not found</div>			

My Profile: MVP

Delete my profile and all associated tasks and log entries

Time Detective

My Profile

First name:

Last name:

Email:

Change Password:

Re-enter Password:

Reminders: ☐ 25 minute ☐ 50 minute

Last login:

Date created:

Time Detective 2021

prompt: Are you sure you want to delete your account and all associated data?

Delete to `/user/{user_id}`

Redirected to home page

success message: Your account was successfully deleted. → `/index`

error message: User not found

Page	HTTP Verb	API Endpoint	Description	CRUD Operation
My Profile /profile	DELETE	/api/user/{ user_id }	Delete a user and all associated task and task log records.	DELETE
Can only be done by a logged in user.	Parameters:			
	user_id			
	Response:			
	Code: 200 Description: Successful operation			
	Code: 404 Description: User not found			

Dashboard: MVP

Return a summary of all tasks, number of times the task was completed, and cumulative time spent on the task

The screenshot shows a web browser window titled "Your Dashboard on Time Detective" with the URL "http://timedetective.io/dashboard". The page has a header with "Time Detective" and navigation links for "Manage Users", "Manage Tasks", "Task Tracker", and "My Profile / Dashboard / Logout". The main content area is titled "My Dashboard" and contains a table with the following data:

Task Name	Quantity	Cumulative Time	Last Logged
xxxxxxxxxx	3	00:34:53	03/13/2021
xxxxxxxxxx	8	03:53:34	03/13/2021
xxxxxxxxxx	10	13:13:28	03/13/2021
xxxxxxxxxx	25	02:34:43	03/14/2021

Below the table, a note states: "Manage Users" button only present on users with Admin role.

→ GET to /logs/summary

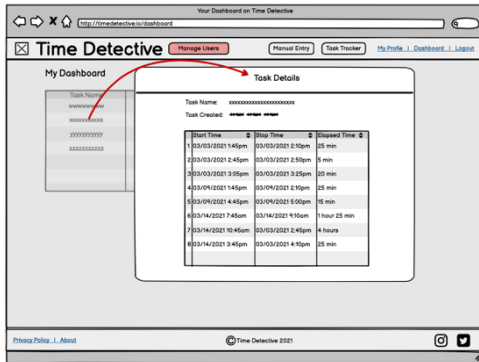
← Task summary returned to page

error message: No tasks found. Add a task.
error message: No logs found. Log a task.

Page	HTTP Verb	API Endpoint	Description	CRUD Operation
Dashboard /dashboard	GET	/logs/summary	Summary of all tasks logged by user	SELECT
<div>Parameters: user_id</div>				
<div>Can only be done by a logged in user.</div>				
<div>Response: Code: 200 Description: Successful operation <pre>[{ "task_id": 0, "task_name": "task_name", "number_of_entries": 1, "cumulative_time_spent_on_task": 1 }, ...]</pre> Code: 400 Description: Invalid user id supplied Code: 404 Description: Tasks not found</div>				

Dashboard: MVP

Return a list of all log entries for an individual task



GET to `/logs/{task_id}`

Individual entries for selected task are returned to the page

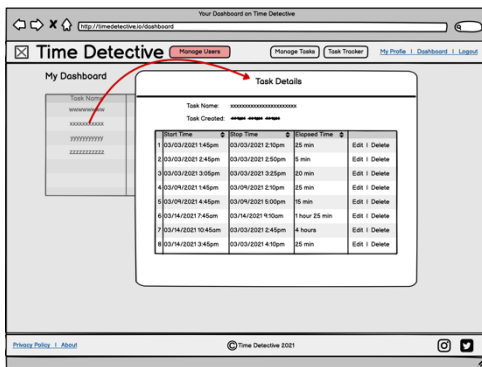
error message: No tasks found. Add a task.
error message: No logs found for this task. Log a task.

Page	HTTP Verb	API Endpoint	Description	CRUD Operation
Dashboard /dashboard	GET	<code>/logs/{task_id}</code>	List of all log entries for a task	SELECT
Parameters: task_id				
Response: Code: 200 Description: Successful operation <pre>{ "task_id": 0, "task_name": "task_name", "start_time": timestamp, "end_time": timestamp, "elapsed_time": 1 }, ...]</pre> Code: 400 Description: Invalid task id supplied Code: 404 Description: Tasks not found				

Can only be done by a logged in user.

Dashboard: MVP

Edit an individual log entry for a task



GET to /log/{log_id}

Individual log entry is returned to the page

error message: Log not found.

PUT to /log/{log_id}

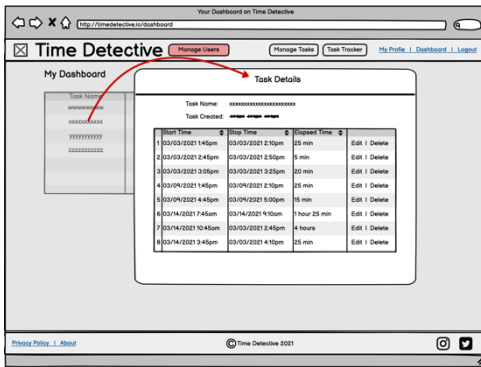
Returned to the log list for an individual task page

error message: Required data is missing.

Page	HTTP Verb	API Endpoint	Description	CRUD Operation
Task Tracker /tracker	GET	/log/{log_id}	Get a single log entry	SELECT
Parameters:				
log_id				
Can only be done by a logged in user.				
Response:				
Code: 200 Description: Successful operation				
<pre>{ "log_id": 0, "task_id": 0, "user_id": 0, "task_name": "task_name", "start_time": timestamp, "end_time": timestamp, "elapsed_time": 1 }</pre>				
...				
Code: 400 Description: Invalid log id supplied				
Code: 404 Description: Log entry not found				
Page	HTTP Verb	API Endpoint	Description	CRUD Operation
Task Tracker /tracker	PUT	/log/{log_id}	Update a log entry from form data	UPDATE
Parameters:				
task id				
Body:				
<pre>{ "log_id": 0, "task_id": 0, "user_id": 0, "task_name": "task_name", "start_time": timestamp, "end_time": timestamp, "elapsed_time": 1 }</pre>				
Response:				
Code: 200 Description: Successful operation				
Code: 400 Description: Invalid task_id supplied				
Code: 404 Description: Task not found				

Dashboard: MVP

Delete an individual log entry for a task



prompt: Are you sure you want to delete this log entry?

DELETE to `/log/{log_id}`

Returned to the log list for an individual task page

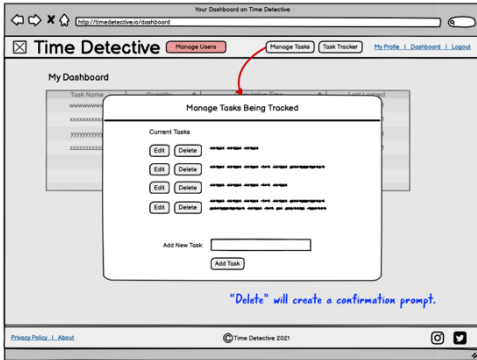
error message: Log entry not found.

Page	HTTP Verb	API Endpoint	Description	CRUD Operation
Dashboard /dashboard	GET	<code>/log/{task_id}</code>	List of all task entries logged by user	SELECT
<div>Parameters:</div> <div>task_id</div>				
<div>Response:</div> <div> Code: 200 Description: Successful operation <pre> { "task_id": 0, "task_name": "task_name", "start_time": timestamp, "end_time": timestamp, "elapsed_time": 1 }, ...] </pre> </div> <div> Code: 400 Description: Invalid task id supplied </div> <div> Code: 404 Description: Tasks not found </div>				

Can only be done by a logged in user.

Dashboard: MVP

Return a list of tasks being tracked by the user



GET to /tasks

List of tasks are returned to the page

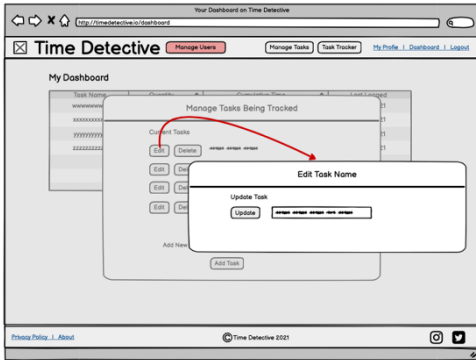
error message: No tasks found.
error message: Required data missing.

Page	HTTP Verb	API Endpoint	Description	CRUD Operation
Dashboard /dashboard	GET	/tasks	List of all tasks being tracked by the user	SELECT
Parameters:				
Response:				
Code: 200 Description: Successful operation				
<pre>[{ "task_id": 0, "task_name": "task_name", "start_time": timestamp, "end_time": timestamp, "elapsed_time": 1 }, ...]</pre>				
Code: 404 Description: Tasks not found				

Can only be done by a logged in user.

Dashboard: MVP

Edit a tracked task



GET to /task/{task_id}

Task is returned to the page

error message: Task not found.

PUT to /task/{task_id}

Redirected to the task list

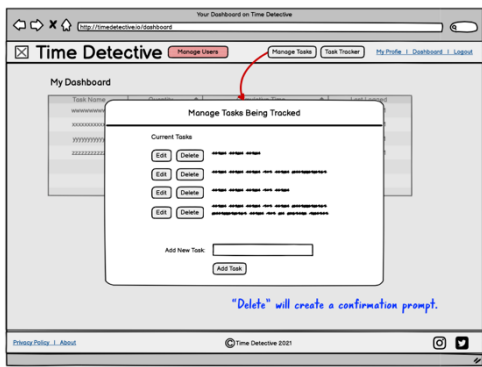
error message: Required data missing.

Page	HTTP Verb	API Endpoint	Description	CRUD Operation
Dashboard /dashboard	GET	/task/{task_id}	List a single task saved by a user	SELECT
Can only be done by a logged in user.	Parameters:			
	task_id			
	Response:			
	Code: 200 Description: Successful operation			
	<pre>{ "task_id": 0, "task_name": "task_name", "task_description": "task_description", "date_created": date }</pre>			
	Code: 400 Description: Invalid task id supplied			
	Code: 404 Description: Task not found			

Page	HTTP Verb	API Endpoint	Description	CRUD Operation
Dashboard /dashboard	PUT	/task/{task_id}	Update task details from form data	UPDATE
Can only be done by a logged in user.	Parameters:			
	task_id			
	Body:			
	<pre>{ "task_id": 0, "task_name": "user_first_name", "task_description": "task_description", "date_created": date }</pre>			
	Response:			
	Code: 200 Description: Successful operation			
	Code: 400 Description: Invalid task_id supplied			
	Code: 404 Description: Task not found			

Dashboard: MVP

Delete an existing task being tracked by the user



prompt: Are you sure you want to delete this task?

DELETE to /task/{task_id}

List of tasks are returned to the page

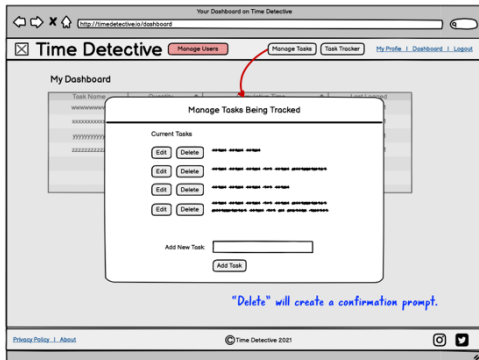


success message: Task successfully deleted.

Page	HTTP Verb	API Endpoint	Description	CRUD Operation
Dashboard /dashboard	DELETE	/task/{ task_id }	Delete a task	DELETE
Can only be done by a logged in user.	Parameters:			
	task id			
	Response:			
	Code: 200 Description: Successful operation			
	Code: 400 Description: Invalid task_id supplied			
	Code: 404 Description: Task not found			

Dashboard: MVP

Create a new task to be tracked by the user



POST to /task

List of tasks are returned to the page

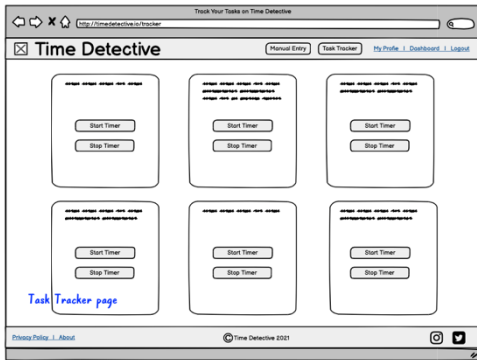
success message: Task successfully added.

error message: Required data missing.

Page	HTTP Verb	API Endpoint	Description	CRUD Operation
Dashboard /dashboard	POST	/task	Create a new task to track	CREATE
Can only be done by a logged in user.	Parameters:			
	Body:			
	Form data:			
	<ul style="list-style-type: none">task_nametask_description			
	Response:			
	Code: 200 Description: Successful operation			
	Code: 405 Description: Invalid input			

Task Tracker: *MVP*

View all tasks to be tracked by the user as a “timer”.



GET to /tasks

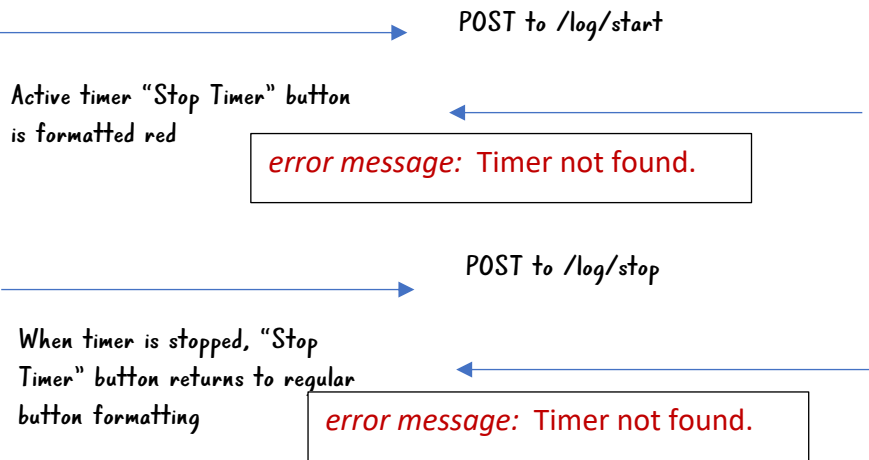
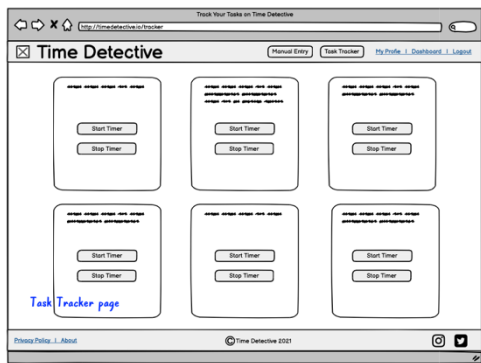
List of tasks are returned to the page and formatted as 'timers'



SEE PAGE 12.

Task Tracker: MVP

Start and stop a timer

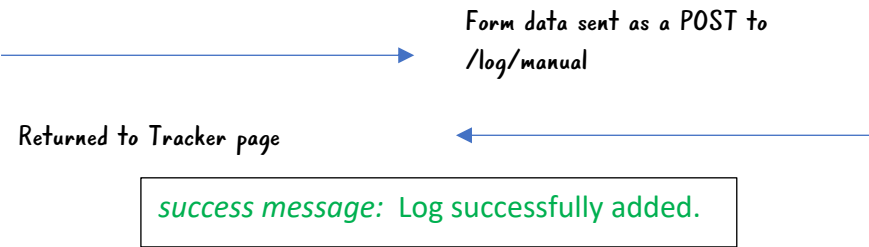
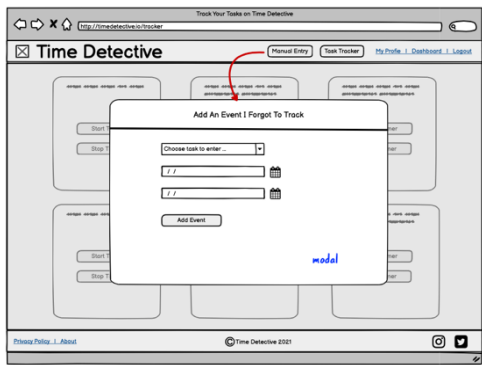


Page	HTTP Verb	API Endpoint	Description	CRUD Operation
Task Tracker /tracker	POST	/log/start	Create a new entry in the task tracking log when a timer starts.	CREATE
Can only be done by a logged in user.	Parameters:			
	Body:			
	Timer data:			
	<ul style="list-style-type: none">• task_id• task_name• start_time			
	Response:			
	Code: 200 Description: Successful operation			
	Code: 405 Description: Invalid input			

Page	HTTP Verb	API Endpoint	Description	CRUD Operation
Task Tracker /tracker	POST	/log/stop	Update an entry in the task tracking log when a timer stops	CREATE
Can only be done by a logged in user and only if a timer has already started.	Parameters:			
	Body:			
	Timer data:			
	<ul style="list-style-type: none">log_idend_time			
	Response:			
	Code: 200 Description: Successful operation			
	Code: 405 Description: Invalid input			

Task Tracker: MVP

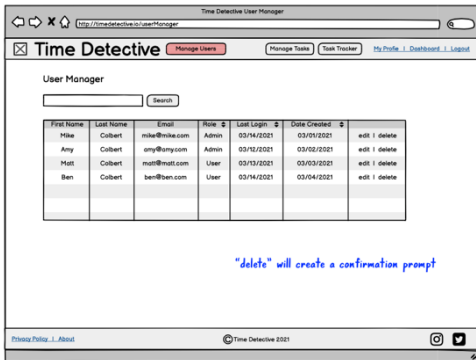
Manually enter a timer event in the log if you forgot to start a timer.



Page	HTTP Verb	API Endpoint	Description	CRUD Operation
Task Tracker /tracker	POST	/log/manual	Create a new, manual entry in the task tracking log from a web form.	CREATE
Can only be done by a logged in user.	Parameters:			
	Body: Form data: <ul style="list-style-type: none">task_idstart_timeend_time			
Response: Code: 200 Description: Successful operation Code: 405 Description: Invalid input				

User Manager: MVP

List all users registered in the application



GET to /admin/users

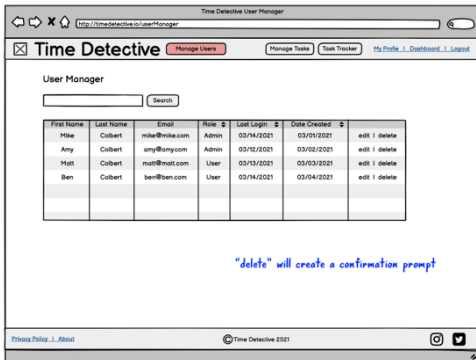
Return list of all users

error message: Please log in as an administrator.

Page	HTTP Verb	API Endpoint	Description	CRUD Operation
User Manager /userManager	GET	/admin/users	List of all users registered on the application	SELECT
<div>Parameters: None</div>				
<div>Can only be done by a logged in with an administrator role.</div>				
<div>Response: Code: 200 Description: Successful operation <pre>[{ "user_id": 0, "first_name": "user_first_name", "last_name": "user_last_name", "email": "user_email_address", "role": 1, "password": "hashed_password", "date_created": date, "last_login": date }, ...]</pre> Code: 403 Description: Forbidden</div>				

User Manager: MVP

Delete a user and all associated tasks and log entries from the application



prompt: Are you sure you want to delete this user and all associated data?

DELETE to `/admin/user/{user_id}`

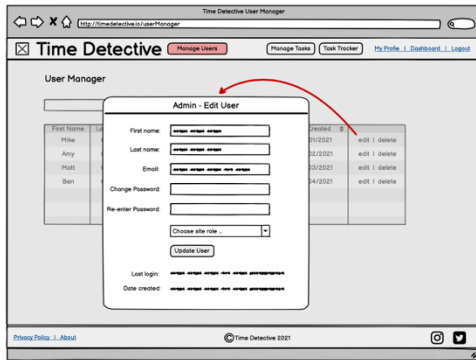
Redirected to list of all users

success message: User successfully deleted.

Page	HTTP Verb	API Endpoint	Description	CRUD Operation
User Manager /userManager	DELETE	/admin/user/{ user_id }	Delete a user	DELETE
Can only be done by a logged in with an administrator role.	Parameters:			
	user id			
	Response:			
	Code: 200 Description: Successful operation			
	Code: 400 Description: Invalid task_id supplied			
	Code: 403 Description: Forbidden			
Code: 404 Description: Task not found				

User Manager: MVP

Update a registered user, including changing the user's role



GET to /admin/user/{user_id}

Returns an individual user to the page

error message: User not found.

PUT to /admin/user/{user_id}

Redirected to list of all users

Page	HTTP Verb	API Endpoint	Description	CRUD Operation
User Manager /userManager	GET	/admin/user/{user_id}	List a single user	SELECT
<p>Can only be done by a logged in with an administrator role.</p>				
<p>Parameters:</p> <p>user_id</p>				
<p>Response:</p> <p>Code: 200 Description: Successful operation</p> <pre>{ "user_id": 0, "first_name": "user_first_name", "last_name": "user_last_name", "email": "user_email_address", "role": 1, "password": "hashed_password", "date_created": date, "last_login": date }</pre> <p>Code: 400 Description: Invalid user id supplied</p> <p>Code: 403 Description: Forbidden</p> <p>Code: 404 Description: User not found</p>				

success message: User updated successfully.

error message: This email address is already registered.

error message: Passwords do not match.

error message: Required data is missing.

Page	HTTP Verb	API Endpoint	Description	CRUD Operation
User Manager /userManager	PUT	/admin/user/{user_id}	Update user details from form data	UPDATE
<p>Can only be done by a logged in user.</p>				
<p>Parameters:</p> <p>task id</p>				
<p>Body:</p> <pre>{ "user_id": 0, "first_name": "user_first_name", "last_name": "user_last_name", "email": "user_email_address", "role": 1, "password": "hashed_password", "date_created": date, "last_login": date }</pre>				
<p>Response:</p> <p>Code: 200 Description: Successful operation</p> <p>Code: 400 Description: Invalid task_id supplied</p> <p>Code: 403 Description: Forbidden</p> <p>Code: 404 Description: Task not found</p>				

Set Pomodoro Timer Length: *Stretch Goal*

User can choose 25- or 50-minute timer lengths.

→ PUT to `/user/{user_id}`

← Redirected to "My Profile" page

success message: Your account was successfully updated. → /myprofile

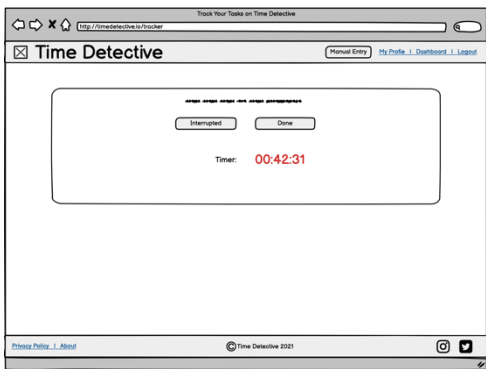
error message: This email address is already registered.

error message: Passwords do not match.

error message: Required data is missing.

Page	HTTP Verb	API Endpoint	Description	CRUD Operation
My Profile /profile	PUT	/user/{ user_id }	Update user details in database by user id	UPDATE
Use userID 1 for testing. Can only be done by a logged in user.	Parameters: Body: <pre> { "user_id": 0, "first_name": "user_first_name", "last_name": "user_last_name", "email": "user_email_address", "password": "plain text_password", "password2": "plain text_password", "archive": 0 } </pre>			
	Response: Code: 200 Description: Successful operation Code: 400 Description: Invalid user id supplied Code: 404 Description: User not found			

User Was Interrupted: *Stretch Goal*
User can log when they are interrupted during a task.



→ POST to /log/interrupt/{task_id}

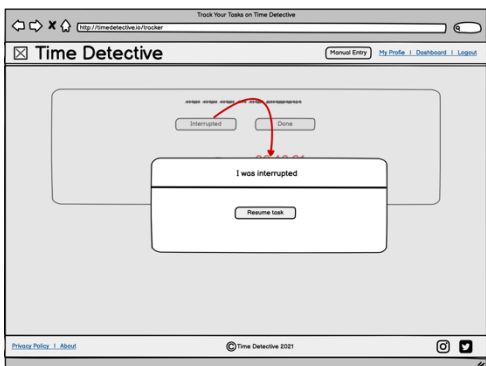
“Resume” modal appears ←

error message: Task not found.

Page	HTTP Verb	API Endpoint	Description	CRUD Operation
Task Tracker /tracker	POST	/log/interrupt/{task_id}	Create a new entry in the task tracking log when a timer starts.	CREATE
Can only be done by a logged in user.	Parameters:			
	Body:			
	Timer data: <ul style="list-style-type: none">task_idtimeinterrupted			
Response:				
Code: 200 Description: Successful operation				
Code: 405 Description: Invalid input				

User Resumes Working on Task: *Stretch Goal*

User can log when they have resumed working on a task after an interruption.



→ POST to /log/resume/{task_id}

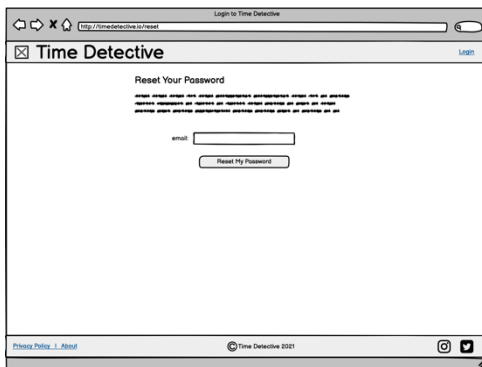
Returns to task timer screen

error message: Task not found.

Page	HTTP Verb	API Endpoint	Description	CRUD Operation
Task Tracker /tracker	POST	/log/resume/{task_id}	Create a new entry in the task tracking log when a timer starts.	CREATE
Can only be done by a logged in user.	Parameters:			
	Body:			
	Timer data:			
	<ul style="list-style-type: none">• task_id• time• resume			
	Response:			
	Code: 200 Description: Successful operation			
	Code: 405 Description: Invalid input			

Password Reset: *Stretch Goal*

User can reset a lost or forgotten password.



POST to /user/reset

Password reset link is emailed

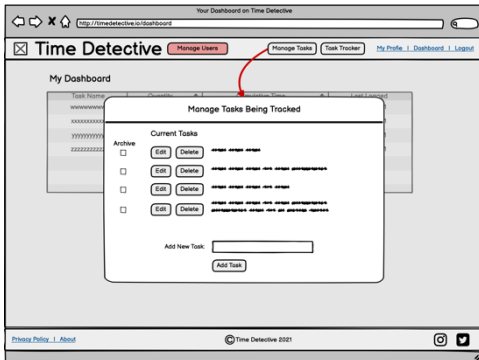
success message: Password reset link has been emailed to you.

error message: Email not found.

Page	HTTP Verb	API Endpoint	Description	CRUD Operation
Task Tracker /tracker	POST	/user/reset/{user_id}	Create a new entry in the task tracking log when a timer starts.	CREATE
Can only be done by a logged in user.	Parameters:			
	Body: User data: <ul style="list-style-type: none">email			
	Response:			
	Code: 200 Description: Successful operation			
	Code: 405 Description: Invalid input			
	Code: 404 Description: Email not found			

Archive A Task: *Stretch Goal*

User can archive a task, so it doesn't appear in active list of tasks, but is available for future reports and analytics.



→ GET to `/task/{task_id}`

Task is returned to the page

error message: Task not found.

→ PUT to `/task/{task_id}`

Redirected to the task list

error message: Required data missing.

Page	HTTP Verb	API Endpoint	Description	CRUD Operation
Dashboard <code>/dashboard</code> <i>Can only be done by a logged in user.</i>	PUT	<code>/task/{task_id}</code>	Update task details from form data	UPDATE
Parameters: task id				
Body: <pre>{ "task_id": 0, "task_name": "user_first_name", "user_id": 0, "date_created": date, "archive": 0 }</pre>				
Response: Code: 200 Description: Successful operation Code: 400 Description: Invalid task_id supplied Code: 404 Description: Task not found				