Sereni App Documentation

Release latest

Sarah W

Sep 29, 2022

INSTALLATION

This documentation will help you learn all about creating projects, sending invoices, receiving payments, creating tasks, billing projects, recurring invoices and much more.

Note: Sereni is build with Laravel Framework

To purchase Sereni Invoicing app Click Here

CHAPTER

ONE

INSTALL

Thanks for purchasing Sereni.

Note: Build with Laravel 9 Framework

Note: Requires PHP >= 8.0 and MySQL.

Attention: You can check your server requirements by downloading this file extract it and upload to your server and access it via a URL i.e http://yourdomain.com/requirements.php (Delete after use).

1.1 Required PHP Extensions

- OpenSSL
- PDO
- Mbstring
- Tokenizer
- JSON
- CURL
- Mysqli
- Zip
- GD

Release Notes: Changelogs

Any feature you need to see on Sereni? request it here

1.2 Create Database

Attention: Do not use a password that contains a #(Hash) character (It will be treated as a comment)

You'll need to create a new database along with a user to access it. Most hosting companies provide an interface to handle this or you can run the SQL statements below.

- Give your database a name e.g **sereni**
- Create a database user and set up a password
- Add the user to the database and give the user **All privileges** to the database

```
CREATE DATABASE sereni;
CREATE USER 'sereni' IDENTIFIED BY 'sereni@@';
GRANT ALL PRIVILEGES ON sereni.* to sereni@'%' identified by 'sereni@@';
FLUSH PRIVILEGES;
```

- You will need to enter this credentials in the steps below.

There are 2 steps to install Sereni

- Using the built in web installer
- Laravel artisan install method (Requires terminal)

Attention: Do not modify config files located in /config folder incase you need to modify configuration paramers do it in .env file (Located in application's root folder). When you modify /config files your changes may be overwritten during an update.

1.3 Web Installer

Download the code from Envato Market Downloads page. The zipped file includes all dependencies. The following steps shows how to install Sereni;

The steps below shows how you can install sereni to be accessible via a sub domain e.g **https://portal.yourdomain.com**

- Create a folder inside **public_html or www** e.g **sereni** and upload your downloaded files into the folder **/public_html/sereni** (Make sure it includes hidden files e.g .env)
- Go to **sub domains** on your cpanel dashboard and create a sub domain
- Enter your preferred sub domain e.g portal.yourdomain.com
- Enter **/public_html/sereni/public** as your **Document Root**
- Save it and access **http://portal.yourdomain.com/installer**

Attention: The url **/installer** does not point to a folder so do not attempt to create a folder named installer

If you encounter an issue displaying images refer to troubleshooting tips below.

1.3.1 Installation

Once you can access the site the initial setup screen will enable you to configure the database as well as create the initial admin user. The first page of the web installer checks if your server meets the requirements to run the application.

> **Attention:** Sereni requires PHP 8.0+

Click **Next** if everything is alright if an extension is missing please contact your hosting provider or install it. The next step checks **directory permissions**. The folders listed should be writable please do NOT set your permissions to **777**. The next step requires database and account information. Enter the information and complete the installation.

> **Attention:** You will need to setup CRON to run every minute as shown below otherwise recurring invoices or tasks will not work.

1.3.2 File Permissions

The webserver should be able to write to this directories **storage**, **public** and **bootstrap/cache**. Here is a sample of how you can set the permissions in ubuntu server.

```
sudo chown -R ubuntu:www-data /path/to/sereni
cd /path/to/sereni
sudo find -type f -exec chmod 664 {} \;
sudo find -type d -exec chmod 775 {} \;
sudo chgrp -R www-data bootstrap/cache storage
sudo chmod -R ug+rwx bootstrap/cache storage
```

- Enter your application name and application URL (e.g https://portal.yourdomain.com)
- Enter your database access information that you used when creating database.
- Enter your admin account information. (This is the admin account you are going to login with)
- Click on install and Sereni will perform the migrations and seeding.
- If everything went well, you should be redirected to login page and you can login using admin account you created above.

> **Attention:** You will need to setup email inorder to verify users accounts. More on that in next article (Configure)

1.4 Installing through SSH (Artisan command)

If you need to install the app using `php artisan` command proceed as follows;

- Open **.env** file and update your database credentials i.e **DB_HOST=xxxxx, DB_PORT=3306, DB_DATABASE=xxxx, DB_USERNAME=xxxx, DB_PASSWORD=xxxx** (You can change other configurations later).
- Run command `php artisan sereni:install` to start the installation.
- The app will run migrations and you will be asked to enter admin name, email and password.

- After successfull install you can now access your dashboard using http://portal.yourdomain.com
- Use your admin account to login.

Note: Admin account created using `php artisan sereni:install` command does not require email verification.

1.5 Email Configuration

- Sereni supports SMTP, Mailgun, Postmark, Amazon SES, and sendmail.
- If you have no idea how to configure email sending, read on the next guide **Configuration**.
- For more information check https://discuss.beanflare.com

1.6 CRON Configuration

Add a CRON job as shown below;

```
* * * * * cd /path/to/sereni && php artisan schedule:run >> /dev/null 2>&1
```

This Cron will call the command scheduler every minute. When the **schedule:run** command is executed, Sereni will evaluate your scheduled tasks and runs the tasks that are due.

More information available here https://discuss.beanflare.com

1.7 Queue Configuration (optional)

Note: For VPS or AWS EC2 users, we recommend installing Supervisord to monitor your processes. Steps on how to install Supervisor on ubuntu are described below

If you need to use supervisord to monitor your queued jobs follow the steps below;

- Open **app/Console/Kernel.php** and comment the line `$schedule->command('queue:work --queue=default,high,normal,low --tries=3')....`
- Now install and start supervisor as described below;

1.7.1 Installing Supervisor

Supervisor is a process monitor for the Linux operating system, and will automatically restart your queue:work process if it fails. To install Supervisor on Ubuntu, you may use the following command:

```
sudo apt-get install supervisor
```

Supervisor configuration files are typically stored in the **/etc/supervisor/conf.d** directory. Within this directory, you may create any number of configuration files that instruct supervisor how your processes should be monitored. For example, let's create a sereni-worker.conf file that starts and monitors a queue:work process:

```
[program:sereni-worker]
process_name=%(program_name)s_%(process_num)02d
command=php /path/to/sereni/artisan queue:work --queue=default,high,normal,low --tries=3
autostart=true
autorestart=true
user=ubuntu
numprocs=1
redirect_stderr=true
stdout_logfile=/path/to/sereni/worker.log
```

You can refer to laravel docs

1.7.2 Starting Supervisor

Once the configuration file has been created, you may update the Supervisor configuration and start the processes using the following commands:

`sudo supervisorctl reread`

`sudo supervisorctl update`

`sudo supervisorctl restart all`

For more information on Supervisor, consult the Supervisor documentation.

See the details here for additional configuration options.

1.8 Troubleshooting

- Check your webserver log (ie, /var/log/apache2/error.log) and the application logs (storage/logs/laravel-error.log) for more details or set `APP_DEBUG=true` in .env
- Getting 404 not found when i access http://portal.mydomain.com/installer - Ensure your sub domain ROOT Document points to /path/to/sereni/public folder and not /path/to/sereni folder.
- I cannot see a folder named **installer** - The url /installer is a laravel route and not a folder. You will be redirected to /installer if the application detects that the app needs to be installed.
- To resolve `file_put_contents(...): failed to open stream: Permission denied` run `chmod -R 777 storage` then `chmod -R 755 storage`
- Running `composer install --no-dev` and `composer dump-autoload` can sometimes help with composer problems.
- Getting error message "Database connection/migration failed" all database credentials are correct. Check that your database user has enough privileges to perform database actions, sereni database should be empty or your password contains a #(Hash).
- Composer install error: `Fatal error: Allowed memory size of...` Try the following: `php -d memory_limit=128M /usr/local/bin/composer install --no-dev`
- My CRONs are not running and i get an error **ErrorException with message 'Invalid argument supplied for foreach()' in /home/project/vendor/symfony/console/Input/ArgvInput.php** to fix this, enter your CRON to run every minute as shown `php -d register_argc_argv=On /path/to/sereni/artisan schedule:run >/dev/null`

CHAPTER

TWO

CONFIGURE

You can check .env.example file to see additional settings.

2.1 CRON Setup

Create one CRON that runs **every minute** as follows;

```
* * * * * php /path-to-your-project/artisan schedule:run >/dev/null
```

2.2 Email Queues

Queues allow you to defer the processing of a time consuming task, such as sending an email, until a later time. Deferring these time consuming tasks drastically speeds up web requests to your application. You can change the queue driver in your .env file `QUEUE_DRIVER=database`

Note: You can process the jobs by running `php artisan queue:work --queue=default,high,normal,low --tries=1`.

2.2.1 Supervisord Configuration

We highly recommend configuring supervisord to monitor your processes and restart your queue if it fails. Read more on Supervisord Configuration

2.3 Application Configuration

You can set your application to production by modifying this value in .env file.

- `APP_ENV` - Set it to **production**
- `APP_DEBUG` - Set it to **false**

2.4 System Configuration

2.4.1 Backup Settings

- `BACKUP_DISKS` - Comma separated list of backup disks e.g **local,s3**
- `BACKUPS_MAIL_ALERT` - Email to send notifications on successfull backup
- `BACKUPS_SLACK_WEBHOOK` - Slack webhook to post backup notifications
- `BACKUPS_SLACK_CHANNEL` - Slack channel to post backup notifications. Default blank

2.4.2 Email Settings

- `MAIL_DRIVER` - Email driver. Default **smtp**
- `MAIL_HOST` - Email host e.g **smtp.mailtrap.io**
- `MAIL_PORT` - Outgoing email port e.g **2525**, **587**, or **465**
- `MAIL_USERNAME` - Outgoing email username
- `MAIL_PASSWORD` - Outgoing email password
- `MAIL_ENCRYPTION` - Default null other options **tls**, **null**, or **ssl**
- `MAIL_FROM_ADDRESS` - The email address that sends emails (Your company address)
- `MAIL_FROM_NAME` - Your company name that appears on emails

> **Attention:** To send a test email go to **Settings** -> **System Settings** and click on **Test Email** button.

2.4.3 Paypal Configuration

To setup paypal;

- Login to Paypal account
- Create a REST API app
- Copy Client ID and paste it in your .env file PAYPAL_CLIENT_ID=your-key
- Copy Secret key and paste in your .env file PAYPAL_CLIENT_SECRET=your-secret
- At the bottom of the page click Add Webhook
- Set the webhook URL to https://{YOUR-DOMAIN}/api/v1/paypal/webhook replace {YOUR-DOMAIN} with your actual domain e.g https://portal.domain.com/api/v1/paypal/webhook
- Enable events **Payment Authorization created, Payment authorization voided, Payment capture completed, Payment capture denied, Payment Capture pending, Payment capture refunded, Payment capture reversed, Payment order cancelled, Payment order created**
- After the webhook is created, copy the Webhook ID and paste to your .env file PAYPAL_WEBHOOK_ID={your-webhook-id}

> **Attention:** To enable PayPal Live, open .env file and change PAYPAL_MODE=sandbox to PAYPAL_MODE=live

2.4.4 Stripe Configuration

To configure Stripe, proceed as follows;

- Login to your stripe dashboard account
- Get your API Keys by clicking Developers section

Open your .env file in Sereni and modify the values below;

```
STRIPE_KEY={YOUR_STRIPE_PUBLISHABLE_KEY}
STRIPE_SECRET={YOUR_STRIPE_SECRET_KEY}
```

2.4.5 Stripe Webhook Configuration

To handle Stripe webhooks, proceed as follows;

- Login to your stripe dashboard and click on Developers section.
- Click Webhooks -> Add Endpoint button
- Enter webhook URL as https://{YOUR-DOMAIN}/api/v1/stripe/webhook replace {YOUR-DOMAIN} with your actual domain e.g https://portal.domain.com/api/v1/stripe/webhook
- Enable events **payment_intent.canceled, payment_intent.created,payment_intent.succeeded, payment_intent.processing, payment_intent.payment_failed**
- Save and copy your Signing Secret and paste in your .env file STRIPE_WEBHOOK_SECRET={YOUR_STRIPE_WEBHOOK_KEY}

2.4.6 Razorpay Configuration

To configure RazorPay, proceed as follows;

- Login to your razorpay dashboard account
- Get your API Keys by clicking Settings -> API Keys section

Open your .env file and modify the values below;

```
RAZORPAY_KEY={RAZORPAY_KEYID}
RAZORPAY_SECRET={RAZORPAY_SECRET}
```

> **Attention:** Create Razorpay webhook and enter webhook URL as https://{YOUR-DOMAIN}/api/v1/razorpay/webhook replace {YOUR-DOMAIN} with your actual domain e.g https://portal.domain.com/api/v1/razorpay/webhook

2.4.7 Mollie Configuration

To configure mollie, proceed as follows;

- Login to your Mollie dashboard account
- Get your API Keys by clicking on Developers section

Open your .env file and modify the values below;

```
MOLLIE_KEY={MOLLIE_API_KEY}
```

2.5 Google ReCaptcha

To enable recaptcha V3, first get your recaptcha key and secret from Google. Open **.env** file on the ROOT folder and enter your values as shown.

```
RECAPTCHAV3_SITEKEY={your-site-key}
RECAPTCHAV3_SECRET={your-secret}
```

Go to **Settings > System Settings** then enable **Enable Recaptcha**.

CHAPTER

THREE

SOCIAL LOGINS

In addition to typical, form based authentication, Sereni also provides a simple, convenient way to authenticate with OAuth providers.

Tip: Currently supports authentication with **Facebook, Twitter, LinkedIn, Google, GitHub and GitLab**

Attention: Before using Social Logins, you will have to enable it in **Settings > System Settings**. Set **Social Login** to Yes.

Note: New user accounts will be created if they do not exist.

3.1 Facebook Configuration

- Start by creating a Facebook Application on the Developers Console.
- Copy your newly created application **App ID** and **App Secret**
- Open your .env file located in your Root folder and set `FACEBOOK_CLIENT_ID={YOUR-APP-ID}` and `FACEBOOK_CLIENT_SECRET={YOUR-APP-SECRET}`.
- Still in your facebook developer console, under **Valid OAuth Redirect URIs** enter your redirect url as `https://your-domain.com/callback/facebook`. Example; If you have installed sereni in https://portal.domain.com then the callback url will be `https://portal.domain.com/callback/facebook`.
- Now your users can login using Facebook.

3.2 Twitter Configuration

- Start by creating a Twitter Application on the Developers Console.
- Copy your newly created application Consumer API Keys **API key** and **API secret key**
- Open your .env file located in your Root folder and set `TWITTER_CLIENT_ID={YOUR-API-KEY}` and `TWITTER_CLIENT_SECRET={YOUR-API-SECRET}`.
- Still in your twitter developer console, under **Callback URLs** enter your redirect url as `https://your-domain.com/callback/twitter`. Example; If you have installed sereni in https://portal.domain.com then the callback url will be `https://portal.domain.com/callback/twitter`.

3.3 Google Configuration

- Start by creating Google Application on the Developers Console.
- Copy your newly created application **Client ID** and **Client Secret**
- Open your .env file located in your Root folder and set `GOOGLE_CLIENT_ID={YOUR-CLIENT-ID}` and `GOOGLE_CLIENT_SECRET={YOUR-SECRET-KEY}`.
- Still in your google developer console, under **Authorized redirect URIs** enter your redirect url as `https://your-domain.com/callback/google`. Example; If you have installed sereni in https://portal.domain.com then the callback url will be `https://portal.domain.com/callback/google`.

3.4 Github Configuration

- Start by creating Github Application on the Developers Console.
- Copy your newly created application **Client ID** and **Client Secret**
- Open your .env file located in your Root folder and set `GITHUB_CLIENT_ID={YOUR-CLIENT-ID}` and `GITHUB_CLIENT_SECRET={YOUR-SECRET-KEY}`.
- Still in your github developer console, under **Authorization callback URL** enter your redirect url as `https://your-domain.com/callback/github`. Example; If you have installed sereni in https://portal.domain.com then the callback url will be `https://portal.domain.com/callback/github`.

3.5 LinkedIn Configuration

- Start by creating LinkedIn Application on the Developers Console.
- Copy your newly created application **Client ID** and **Client Secret**
- Open your .env file located in your Root folder and set `LINKEDIN_CLIENT_ID={YOUR-CLIENT-ID}` and `LINKEDIN_CLIENT_SECRET={YOUR-SECRET-KEY}`.
- Still in your linkedin developer console, under **Redirect URLs** enter your redirect url as `https://your-domain.com/callback/linkedin`. Example; If you have installed sereni in https://portal.domain.com then the callback url will be `https://portal.domain.com/callback/linkedin`.

3.6 Gitlab Configuration

- Start by creating Gitlab Application in Applications Settings.
- Enter your **Redirect URI**. Example; If you have installed sereni in https://portal.domain.com then the callback url will be `https://portal.domain.com/callback/gitlab`. (Use 1 line per URI)
- Under **Scopes** check the **API** checkbox to enable it
- Copy your newly created application **Application ID** and **Secret**
- Open your .env file located in your Root folder and set `GITLAB_CLIENT_ID={YOUR-APPLICATION-ID}` and `GITLAB_CLIENT_SECRET={YOUR-SECRET-KEY}`.

CHAPTER

FOUR

UPDATE

Note: We recommend backing up your files and database before updating the app.

Sereni checks for updates daily and notifies you via email if there is a new update. You can disable update notifications in **Settings > Updates > Notify me on new updates**.

Attention: Updates require a valid purchase code. Enter your purchase code in **Settings > Updates** and save.

To update your portal go to **Settings > Updates** and click on **Install** button to install available updates.

Attention: Your application will be set to maintenance mode when performing the update and restored after the update process has completed.

If you're moving servers make sure to copy over the .env file.

If incase the system update fails and shows **Maintenance Mode**, run the command `php artisan up` manually and check the storage/logs folder for issues. You will receive a notification if the update is successful/fails.

4.1 Fixing Permissions

Never set your folder to 777 permission. If you give any of your folders 777 permissions, you are allowing any user to read, write and execute any file in that directory. What this means is you have given any user (any hacker or malicious person in the entire world) permission to upload ANY file, virus or any other file, and THEN execute that file.

To fix permissions issue run command `sudo chown -R my-user:www-data /path/to/your/sereni` The first **my-user** is name of the user, and the second **www-data** is the name of the group.

Then give the webserver the rights to read and write to storage and cache.

```
sudo chgrp -R www-data storage bootstrap/cache
```

```
sudo chmod -R ug+rwx storage bootstrap/cache
```

CHAPTER

FIVE

CHANGELOGS

5.1 Version 0.9.0 - 01 October 2022

- Initial release

CHAPTER

SIX

API

Sereni provides a RESTful API. Below are the API Endpoints you can use;

6.1 Authorization

To access the API, you first need to copy your Token. Go to top right corner next to your profile avatar image then click **API Tokens**. Create a token by giving it a name and choose which actions it can perform. Copy and save the Token (It won't be shown again).

Note: Insert your Token directly into the authorization header on any API requests. Example below;

```
curl -X GET
-H "Authorization: Bearer <bearer>"
-H "Content-Type: application/json"
"https://{endpoint}/api/v1/projects"
```

6.2 Authentication Errors

If the bearer token is invalid or expired you will receive a response with the status code set to **HTTP 401 Unauthorized**.

If the token is valid but you don't have enough scope to perform this request you will receive a response with the status code set to **HTTP 403 Forbidden**.

6.3 HTTP-Codes

Actions and errors yield different HTTP response codes. Please have a look at the expected response codes in the following list.

- `200` Request OK
- `201` New resource created
- `304` The resource has not been changed
- `400` The request parameters are invalid
- `401` The bearer token or the provided api key is invalid
- `403` You do not possess the required rights to access this resource.

- `404` The resource could not be found / is unknown.
- `415` The data could not be processed or the accept header is invalid.
- `422` Could not save the entity
- `500` An unexpected condition was encountered
- `503` The server is not available (maintenance work).

6.4 HTTP-Headers

Attention: You must provide the following headers in each request:

Header	Valid Values	Description
Accept	application/json	API Format
Authorization	Bearer {your-token}	Replace with your Token

6.5 Expenses

6.5.1 `POST` - /api/v1/expenses

Create a new expense

```
POST /api/v1/expenses
Accept: application/json
Authorization: Bearer {your-token}
```

Parameters

Name	Required	Description
amount	required	Expense amount e.g 1500.00
category_id	required	Expense category
expense_date	required	Expense date
vendor	required	Vendor name e.g Apple
invoiced	required	A value 0 or 1
project_id	optional	Associated project ID if any

6.5.2 GET - /api/v1/expenses/{id}

Get expense information

```
GET /api/v1/expenses/{id}
Accept: application/json
Authorization: Bearer {your-token}
```

Sample Response


```
{
      "type": "expenses",
      "id": "97807926684684288",
      "attributes": {
          "id": "97807926684684288",
          "reference": "EXP-0008",
          "name": "Purchase AWS EC2",
          "amount": 20,
          "currency": "USD",
          "vendor": "Amazon",
          "project": {
              "id": "88528805735567360",
              "name": "Artemis III"
          },
          "category": {
              "id": "87137425234726912",
              "name": "Office Rent"
          },
          "client": {
              "id": "87139674321195008",
              "name": "Elon Musk"
          },
          "user": {
              "id": "87137522655825920",
              "name": "Sarah W",
              "email": "sarah@domain.com"
          },
          "invoiced": 0,
          "is_recurring": 0,
          "is_visible": 1,
          "notes": null,
          "expense_date": "2022-09-09T00:00:00+03:00",
          "created_at": "2022-09-27T22:33:46+03:00",
          "updated_at": "2022-09-27T22:51:33+03:00"
      }
  }
```

6.5.3 PUT - /api/v1/expenses/{id}

Update an expense

```
PUT /api/v1/expenses/{id}
Accept: application/json
Authorization: Bearer {your-token}
```

Parameters

Name	Required	Description
amount	required	Expense amount e.g 1500.00
category_id	required	Expense category
expense_date	required	Expense date
billable	optional	Whether the expense is billable. Default 1
notes	optional	Expense notes
project_id	required	Associated project ID if any
vendor	optional	Associated vendor name

6.5.4 DELETE - /api/v1/expenses/{id}

Delete an expense

```
DELETE /api/v1/expenses/{id}
Accept: application/json
Authorization: Bearer {your-token}
```

6.5.5 GET - /api/v1/expenses

Get a list of all expenses

```
GET /api/v1/expenses
Accept: application/json
Authorization: Bearer {your-token}
```

6.5.6 GET - /api/v1/expenses/{id}/comments

Show expense comments

```
GET /api/v1/expenses/{id}/comments
Accept: application/json
Authorization: Bearer {your-token}
```

6.6 Payments

6.6.1 POST - /api/v1/payments

Create a new payment

```
POST /api/v1/payments
Accept: application/json
Authorization: Bearer {your-token}
```

Parameters

Name	Required	Description
invoice_id	required	Invoice ID
payment_date	required	Date when the payment was made
amount	required	Amount of payment made
payment_method	required	Payment method ID
notes	optional	Payment additional notes
currency	optional	Payment Currency
send_email	optional	If an email should be sent to client. Default 1

6.6.2 GET - /api/v1/payments/{id}

Get payment information

```
GET /api/v1/payments/{id}
Accept: application/json
Authorization: Bearer {your-token}
```

Sample Response


```
{
      "type": "payments",
      "id": "97281884052131840",
      "attributes": {
          "id": "97281884052131840",
          "reference": "PAY-20220926-0001",
          "method": "Stripe",
          "amount": 32.38,
          "currency": "USD",
          "invoice": {
              "id": "97576297596850176",
              "reference": "INV-20220927-0001"
          },
          "project": {
              "id": "87139846275076096",
              "name": "Artemis"
          },
```


(continues on next page)

(continued from previous page)


```
        "client": {
            "id": "87139674321195008",
            "name": "Elon Musk"
        },
        "notes": null,
        "meta": {
            "orderId": "7U6455882K178700M",
            "status": "COMPLETED",
            "amount": "32.38",
            "currency_code": "USD",
            "invoice_id": "88355381969031168",
            "time": "2022-09-26T08:43:27Z"
        },
        "status": "pending",
        "provider_id": "7U6455882K178700M",
        "payment_date": "2022-10-08T00:00:00+03:00",
        "created_at": "2022-09-26T11:43:28+03:00",
        "updated_at": "2022-09-27T21:32:11+03:00"
    }
}
```


6.6.3 PUT - /api/v1/payments/{id}

Update a payment

```
PUT /api/v1/payments/{id}
Accept: application/json
Authorization: Bearer {your-token}
```

Parameters

Name	Required	Description
invoice_id	required	Invoice ID
payment_date	required	Date when the payment was made
amount	required	Amount of payment made
payment_method	required	Payment method ID
notes	optional	Payment additional notes
currency	optional	Payment Currency

6.6.4 DELETE - /api/v1/payments/{id}

Delete a payment

```
DELETE /api/v1/payments/{id}
Accept: application/json
Authorization: Bearer {your-token}
```

6.6.5 GET - /api/v1/payments

Get a list of all payments

```
GET /api/v1/payments
Accept: application/json
Authorization: Bearer {your-token}
```

6.6.6 GET - /api/v1/payments/{id}/comments

Show payment comments

```
GET /api/v1/payments/{id}/comments
Accept: application/json
Authorization: Bearer {your-token}
```

6.7 Clients

6.7.1 POST - /api/v1/clients

Create a new client

```
POST /api/v1/clients
Accept: application/json
Authorization: Bearer {your-token}
```

Parameters

Name	Required	Description
name	required	Client Name
email	required	Client email address
phone	optional	Client phone number
street	optional	Street Address
zip_code	optional	Zip Code
city	optional	City
state	optional	State
locale	optional	Preferred locale
country_id	required	Country
tax_number	optional	Client tax number if any
currency_id	required	Preferred currency
notes	optional	Additional notes

6.7.2 GET - /api/v1/clients/{id}

Get client information

```
GET /api/v1/clients/{id}
Accept: application/json
Authorization: Bearer {your-token}
```

Sample Response

```
{
      "type": "clients",
      "id": "87264004728295424",
      "attributes": {
          "id": "87264004728295424",
          "reference": "CUST0002",
          "name": "Acme Limited",
          "email": "info@acme.com",
          "currency": {
              "code": "KES",
              "name": "Kenyan Shilling"
          },
          "contact": {
              "id": "87425478868209664",
              "name": "Allen McCain",
              "email": "allen@acme.com"
          },
          "phone": "0790089890",
          "street": "123 Street",
          "city": "Eldoret",
          "state": "CA",
          "zip_code": "50762",
          "country": {
              "code": "KE",
```

(continues on next page)

(continued from previous page)


```
            "name": "Kenya"
        },
        "locale": "en",
        "tax_number": "A-KRA",
        "photo": "/storage/photos/HgRdM8LP1eBgmTJunpQYgTo6Cf7jnSM92iedsN3t.jpg",
        "expense": 0,
        "balance": 0,
        "paid": 0,
        "billing_street": "123 Street",
        "billing_city": "Eldoret",
        "billing_zip": "23423",
        "billing_state": "RV",
        "billing_country": {
            "code": "KE",
            "name": "Kenya"
        },
        "status": "Active",
        "notes": null,
        "created_at": "2022-08-29T19:16:00+03:00",
        "updated_at": "2022-09-28T09:48:46+03:00"
    }
}
```


6.7.3 PUT - /api/v1/clients/{id}

Update client information

```
PUT /api/v1/clients/{id}
Accept: application/json
Authorization: Bearer {your-token}
```

Parameters

Tip: Same as the create new client API parameters

6.7.4 DELETE - /api/v1/clients/{id}

Delete a client

```
DELETE /api/v1/clients/{id}
Accept: application/json
Authorization: Bearer {your-token}
```

6.7.5 GET - /api/v1/clients

Get a list of all clients

```
GET /api/v1/clients
Accept: application/json
Authorization: Bearer {your-token}
```

6.7.6 GET - /api/v1/clients/{id}/projects

Show client projects

```
GET /api/v1/clients/{id}/projects
Accept: application/json
Authorization: Bearer {your-token}
```

6.7.7 GET - /api/v1/clients/{id}/payments

Show client payments

```
GET /api/v1/clients/{id}/payments
Accept: application/json
Authorization: Bearer {your-token}
```

6.8 Projects

6.8.1 POST - /api/v1/projects

Create a new projects

```
POST /api/v1/projects
Accept: application/json
Authorization: Bearer {your-token}
```

Parameters

Name	Required	Description
name	required	Project Name
client_id	required	Project client ID
start_date	required	Project start date
due_date	required	Project due date
description	optional	Description
hourly_rate	optional	Hourly rate
fixed_price	optional	Fixed Price. E.g 3400.00
notes	optional	Project Notes
color	required	gray,red,orange,yellow,pink,cyan,blue
estimate_hours	optional	Project Estimated hours
billing_method	optional	`hourly_staff_rate, hourly_task_rate, hourly_project_rate, fixed_rate`
status	optional	active, on_hold

6.8.2 GET - /api/v1/projects/{id}

Get project information

```
GET /api/v1/projects/{id}
Accept: application/json
Authorization: Bearer {your-token}
```

Sample Response


```
{
      "type": "projects",
      "id": "88528805735567360",
      "attributes": {
          "id": "88528805735567360",
          "reference": "PRO0004",
          "name": "Artemis III",
          "currency": "USD",
          "client": {
              "id": "87139674321195008",
              "name": "Elon Musk"
          },
          "start_date": "2022-10-04T00:00:00+03:00",
          "due_date": "2022-10-08T00:00:00+03:00",
          "hourly_rate": "60.00",
          "fixed_rate": null,
          "progress": 65,
          "estimate_hours": "72.00",
          "billable_time": "85798.00",
          "unbillable_time": "0.00",
          "unbilled": "43254.00",
```


(continues on next page)

(continued from previous page)


```
        "sub_total": 1409.89,
        "total_expenses": 770,
        "billing_method": "hourly_task_rate",
        "status": "active",
        "notes": null,
        "created_at": "2022-09-02T07:01:52+03:00",
        "updated_at": "2022-09-28T08:22:12+03:00"
    }
}
```


6.8.3 PUT - /api/v1/projects/{id}

Update project information

```
PUT /api/v1/projects/{id}
Accept: application/json
Authorization: Bearer {your-token}
```

Parameters

Tip: Same as the create new project API parameters

6.8.4 DELETE - /api/v1/projects/{id}

Delete project

```
DELETE /api/v1/projects/{id}
Accept: application/json
Authorization: Bearer {your-token}
```

6.8.5 GET - /api/v1/projects

Get a list of all projects

```
GET /api/v1/projects
Accept: application/json
Authorization: Bearer {your-token}
```

6.8.6 GET - /api/v1/projects/{id}/boards

Show available project task boards

```
GET /api/v1/projects/{id}/boards
Accept: application/json
Authorization: Bearer {your-token}
```

6.8.7 GET - /api/v1/projects/{id}/tasks

Show available project tasks

```
GET /api/v1/projects/{id}/tasks
Accept: application/json
Authorization: Bearer {your-token}
```

6.8.8 GET - /api/v1/projects/{id}/expenses

Show project expenses

```
GET /api/v1/projects/{id}/expenses
Accept: application/json
Authorization: Bearer {your-token}
```

6.9 Tasks

6.9.1 POST - /api/v1/tasks

Create a new task

```
POST /api/v1/tasks
Accept: application/json
Authorization: Bearer {your-token}
```

Parameters

Name	Required	Description
project_id	required	Project ID
priority_id	required	Task priority ID
name	required	Task Name
start_date	optional	Task start date
due_date	optional	Task due date
hourly_rate	optional	Hourly rate e.g 30.00
milestone_id	optional	Milestone ID
board_id	required	Task board ID
estimated_hours	optional	Task estimated hours e.g 72
description	optional	Description
visible	required	Hide or show to client
is_recurring	optional	Default 0

6.9.2 GET - /api/v1/tasks/{id}

Get task information

```
GET /api/v1/tasks/{id}
Accept: application/json
Authorization: Bearer {your-token}
```

Sample Response


```
{
      "type": "tasks",
      "id": "90750456141320192",
      "attributes": {
          "id": "90750456141320192",
          "project": {
              "id": "88528805735567360",
              "name": "Artemis III"
          },
          "user": {
              "id": "87137522655825920",
              "name": "Sarah W",
              "email": "sarah@acme.com"
          },
          "board": {
              "id": "91217428461260800",
              "name": "Backlog"
          },
          "priority": {
              "id": "87137426274914304",
              "name": "Medium"
          },
          "name": "Setup Network",
          "start_date": "2022-10-04T00:00:00+03:00",
```


(continues on next page)

(continued from previous page)


```
            "due_date": "2022-10-08T00:00:00+03:00",
            "progress": 50,
            "hourly_rate": "45.00",
            "logged_time": 31,
            "estimated_hours": "24.00",
            "description": null,
            "visible": 0,
            "created_at": "2022-09-08T11:09:54+03:00",
            "updated_at": "2022-09-28T08:22:12+03:00"
        }
    }
```


6.9.3 PUT - /api/v1/tasks/{id}

Update task information

```
PUT /api/v1/tasks/{id}
Accept: application/json
Authorization: Bearer {your-token}
```

Parameters

Tip: Same as the create task API parameters above

6.9.4 DELETE - /api/v1/tasks/{id}

Delete a task

```
DELETE /api/v1/tasks/{id}
Accept: application/json
Authorization: Bearer {your-token}
```

6.9.5 GET - /api/v1/tasks

Get a list of all tasks

```
GET /api/v1/tasks
Accept: application/json
Authorization: Bearer {your-token}
```

6.9.6 POST - /api/v1/tasks/{id}/duplicate

Duplicate a task

```
POST /api/v1/tasks/{id}/duplicate
Accept: application/json
Authorization: Bearer {your-token}
```

Parameters

Name	Required	Description
task_id	required	The task ID
timesheets	required	Set to 1 to duplicate time entries. 0 means ignore
checklists	required	Set to 1 to duplicate checklists. 0 means ignore
comments	required	Set to 1 to duplicate comments. 0 means ignore
attachments	required	Set to 1 to duplicate files. 0 means ignore
assignees	required	Set to 1 to duplicate team members. 0 means ignore

CHAPTER

SEVEN

INTRODUCTION

Sereni is a place for you to store information on the people and clients you interact with as a freelancer and track your outreach efforts.

Note: Sereni is build with Laravel Framework

To purchase Sereni Click Here

7.1 Top Navigation

In your portal account, click your login name next to your avatar in the top navigation bar. The following sections will appear in the dropdown menu:

- Settings - Your Profile settings including GDPR.
- API Tokens - Generate API tokens here

7.2 Dashboard

This is the first page you'll see when you login. It provides a general overview of the invoicing status of your freelance business. The page displays the invoice-specific data that's associated with your business. It also allows you to see the updates at a glance.

The dashboard is designed to offer a simple yet powerful overview of your total business accounts:

- **Paid**: The total amount paid to you in the current month
- **Expenses**: The total amount of expenses in current month
- **Tasks**: Total number of tasks completed in current month
- **Balance**: Total outstanding balances

Tip: If you are being paid in a range of currencies, the dashboard will display the totals in your Base Currency.

Chart Below the stats boxes, you'll see a chart presenting your invoicing data in an easy-to-understand graphical format. The data presented in the chart are total invoiced amount versus revenue collected in the current Year.

Tip: If you are being paid in a range of currencies, the dashboard will display the totals in your Base Currency.

7.3 Recent Payments

The recent payment list is incredibly useful as it presents an up-to-date summary of your recent transactions.

CHAPTER

EIGHT

CUSTOMERS

Your customers are the core of your freelance business. Customers in Sereni will typically hold all information specific to a company that your organisation will have a relationship with. In real world terms an Account may be a business entity that is a qualified Sales Prospect, Customer, Supplier or Re-seller and can be used to track all interactions that take place between these entities and your organisation. The relationship between the account and contact records is one-to-many, such that there can be many contacts associated with a single account.

8.1 Contacts

In Sereni, a Contact is an individual who is typically associated with a client (organisation) or Opportunity (qualified prospect). For example if Apple is the Client, then John Smith, Sales Manager of Apple is the Contact. This module holds all information relating to these individuals and also provides a vantage point for any history relating to a Contact record, for example if they sent you an Email.

8.2 List Customers

The Customers page is a list page that presents a summary of all your customers in a user-friendly table. Think of Customers page as the "central station" of your client activity. Most of your day-to-day invoicing actions can be taken from the various links and buttons that appear on this page. And you can use the Customers page as your starting point to explore more in-depth client information, view client projects and more. Now, we'll take a closer look at the setup of the customers page, and the range of actions available to you on the page.

To view your client list page, go to the main sidebar and click the Customers tab.

8.2.1 Overview

The customers page presents a list summary of all your current customers in a table format. The main elements of the table include:

- **Name:** The name of the client
- **Phone:** Company phone number and contact person
- **Email:** The client email address
- **Balance:** The client's payment balance
- **Location:** The client's addresses

8.3 Create Customer

So, you've taken on a new client? Congratulations!

Your customers list is at the heart of your invoicing activity, so it's really important to maintain current information on all your customers. When you start working with a new client, the first thing you'll need to do is to add the new client by entering their contact information and business details.

When creating and saving a new client to your Client list, make sure to have the relevant, up-to-date information at hand. You are only required to enter the information one time. Sereni automatically tracks all invoicing,projects and payment activity for each client.

Client Creation

To create customers individually.

In the customers section, click (New Customer button) to add a new account.
In the modal form, enter the account details.
Click Save

Tip: The New Customer modal is divided into four sections. Enter the information in the relevant fields.

Let's take a closer look at each section:

- **Customer Information**: Enter details about your client's business/company/organization, including the company name, email, tax number and logo (optional).
- **Shipping Address**: Enter shipping address if required.
- **Billing Address**: Enter your client billing address.
- **Others**: Set the default language for this customer.

8.4 Customer Overview Page

Once you have created an account, you can view the details of the account in the record's details page. The details page of an account presents information related to the account - for example, contacts, projects, invoices, payments in a single location.

Tip: You can view how much the client owes you on the left sidebar of the customer details page.

8.4.1 Invoices Section

The Invoices section shows a list of all the client's invoices and accompanying information.

- **Reference**: The invoice number
- **Amount**: The invoice amount
- **Balance**: The invoice balance
- **Due Date**: The date the payment is due
- **Status**: The status of the invoice (Draft, Not Paid, Sent, Viewed, Paid, Overdue)

Tip: You can also create a new invoice for this client via the Create button that appears at the top right of the Invoices section.

8.4.2 Payments Section

The Payments section shows a list of all the client's payments and accompanying information.

- **Reference**: The reference number of the transaction
- **Method**: The payment method (ie. Paypal, Stripe, Cash, etc)
- **Invoice**: The invoice related to this payment
- **Amount**: The payment amount
- **Date**: The date the payment was made

8.4.3 Projects Section

The projects section shows a list of all the client's projects and accompanying information.

- **Name**: Project name
- **Client**: The client's name
- **Start Date**: The start date of the project
- **Due Date**: The date the project is due
- **Members**: Avatars for project team members
- **Cost**: Total cost of the project
- **Billing Method**: Project billing method i.e hourly_rate, fixed ect

8.4.4 Updating Client

Click on the Edit Customer button, at the top left corner of the page. You will now see a modal form, where you can edit any of the fields.

8.4.5 Deleting the Client

You can also delete the specific client directly from their Client Overview page. Click **More** button and choose Delete. This action will delete the client together with all payments, invoices etc.

CHAPTER

NINE

INVOICES

Invoices allow you to bill a Client for your products and/or services, and help you keep track of your income. Every invoice paid means more revenue coming into your business. Create and send professional invoices to your customers in seconds. Once you've entered the client and tax information, you'll have a range of actions at your fingertips – from saving a draft, to sending the invoice to the client via email, to printing a PDF hard copy.

9.1 Overview

The life of an invoice in Sereni system is made up of a number of stages:

- **Draft/Hidden**: When you've created an invoice, but have not yet sent it.
- **Sent**: You've sent the invoice, but the client has not yet paid.
- **Viewed**: The client has logged in and viewed the invoice.
- **Partial**: The invoice has been partially paid.
- **Paid**: Congratulations! The client has paid the full invoice amount.
- **Not Paid**: The invoice remains unpaid.
- **Overdue**: The invoice has passed its due date.

9.2 Invoices list

On the invoices list page you'll see a table with the columns below;

- **Reference**: The number of the invoice
- **Customer**: The name of the customer
- **Status**: The current status of the invoice (Draft, Sent, Viewed, Partial, Paid, Not Paid, Overdue)
- **Due Date**: The date the payment is due
- **Amount**: The total amount of the invoice
- **Balance**: The amount owed by the customer

9.3 Create Invoice

To create a new invoice, go to the Invoices tab on the main sidebar, and click on New Invoice button. This will open a modal offering a series of text and numerical inputs.

Note that each new invoice you create will be automatically numbered in chronological order. This will ensure your records are kept logical and organized. (You have the option to change the invoice number manually in **Settings** - **Invoice Settings**).

The form contains:

- **Customer**: Select from a list of customers
- **Project**: Whether this invoice is attached to a project.
- **Late Fee**: The percentage or amount to be applied if the invoice is not paid on time.
- **Due Date**: The invoice due date
- **Recurring**: Make this invoice recur or generate a one time invoice.
- **Show shipping**: If enabled, a shipping address will show on invoice page.

Tip: You can create a new client while creating a new invoice. Simply click on new client link, situated next to customer field. A pop-up modal will open, enabling you to complete the new client's details. Then continue creating the invoice for this new client.

- **Notes**: Want to enter information to appear as a footer on the invoice? Enter it here. The text will appear at the bottom of the invoice.

Once you've completed creating your invoice, click on **Save** Button and you'll be redirected to the invoice page where you can enter your products/services to bill your clients.

- **Item/Product**: This is the name of the item you are billing for.
- **Description**: Add more information about the item. This will help the customer better understand the job completed, and is also useful for your own reference.
- **Unit Price**: The amount you charge per unit of items. For example, let's say your item is "1 hour consulting", and you charge $80 for an hour of consulting – that is, for 1 item unit. Then you'll enter 80 in the Unit Price field.
- **Quantity**: The number of units being charged. Continuing the above example, let's say you need to charge for 3 hours of consulting, enter the number 3 in the Quantity field.
- **Tax Rate**: Note: To apply tax to the line item, click on the arrow at the right side of the Tax field and select the relevant tax from the drop-down list.
- **Discount**: This is the discount percentage you need to apply for the particular line item.

Click on **Save** button to save the item.

Beneath and to the right of the line item section, you'll find the Totals section:

- **Payable**: This is the amount due before other figures are taken into calculation, such as Tax etc.
- **Payment Made**: The amount paid to date, including partial payments.
- **Balance**: The final balance owed to you, after taxes, payments have been deducted from the charged amount.

9.4 Invoice Page

- **Make Visible**: Use this button to hide/show invoice to client.
- **Pay**: Click this button to make payment to an Invoice.
- **Send**: Email the invoice directly to the email address specified for the client.
- **More button**: Access additional invoice options including updating, deleting invoice.
- **Mark Sent**: When you mark an invoice as sent, only then is the invoice viewable to the client in the client portal, and the client balance is updated to reflect the invoice amount.
- **Delete**: Click here to delete the invoice. It will be deleted and removed from the Invoices list page.
- **PDF**: Download a PDF version of the invoice.

9.5 Viewed Notification

Know when an invoice is viewed, becomes due, or gets paid, so you can take the right actions to manage your cash flow. Set up invoice reminders to automatically email your customers when payment is due.

9.6 Auto Reminders

Save yourself the time and hassle and automate your client communications! An invoice reminder is an automatic email message to remind your customer that an invoice is coming due or that it is overdue. This is a great way to stay on top of reminding your customers that you should be getting paid soon.

You can have Sereni send reminders that invoices will be due to be paid soon, and we call these **Upcoming Reminders**. You can also have Sereni send reminders that invoices are overdue, and we call these **Overdue Reminders**.

9.6.1 Upcoming Reminders

To send invoice reminders before the due date, modify **Settings > Invoice Settings > Upcoming reminders**. Default is 3 days

- Change to the number of days you want a reminder to be sent before the due date.
- Example; Setting it to 2 days will send invoice reminders 2 days before invoice overdue date.

9.6.2 Overdue Reminders

To enable Invoice Overdue Reminders, go to **Settings > Invoice Settings > Auto remind invoices** and enable it.

Once you've got these reminders set up, you don't have to do anything else; Sereni will continue to send these reminders on the schedule you set until the end of time (or until you get paid, whichever comes first).

Tip: Modify the number of days to send each invoice reminder in **Settings - Invoice Settings** section. You may also set late fee to apply on third reminder.

9.7 Recurring Invoice

As a busy freelancer, you work for a variety of clients. Some jobs are one-off, but others are ongoing, whether on a weekly, monthly or other basis. Sereni recurring invoice feature automatically creates invoices for ongoing jobs, and sends the current invoice to the client on a regular, pre-defined basis. For each recurring job, you only need to set up the procedure once.

To make a invoice recur, edit the invoice and select the **Recurring Invoice** dropdown. You can set it to recur every `week, month, quarter, six months and yearly`. Select the start date and a date when the invoice should stop recurring (End Date).

Tip: To stop a recurring invoice, edit the invoice and change **Recurring invoice** field to **No**.

Tip: Reminders are sent based on the due date of the invoice.

Tip: To disable/enable sending invoices immediately they recur, go to **Settings** - **Invoice Settings** and disable/enable **Email on Recur** checkbox..

CHAPTER

TEN

PAYMENTS

Sereni handles your entire freelance invoicing process - from invoicing your client to receiving payment. What's more, you can receive payments directly and automatically via supported gateways enabling totally smooth management of your customer accounts using your choice of payment provider.

10.1 Payments List

The Payments list page displays a summary of all payments once they have been received. Payments are recorded in two ways:

1. **Automatic payment**: If your client has paid you via any of the supported payment gateways, the payment will be automatically recorded in the Payments list. You will be notified on your dashboard page in the notification section, and also via email.
2. **Manual payment**: If your client has paid you via cash, check, bank transfer, credit card or any other payment system not linked to your portal, you will need to enter the payment manually on the payments page.

Whether automatic or manual entry, the Payments list page presents an overview of all payments received in a user-friendly table format. Now, we'll take you through the various columns in the Payments table from left to right:

- **Reference**: Transaction reference number.
- **Method**: The method of payment used, ie. PayPal, Bank Transfer, etc
- **Invoice**: Invoiced linked to this payment
- **Amount**: The payment amount that was received
- **Payment Date**: The date the payment was received

10.2 Refunding a Payment

If you need to refund a payment, go to the relevant payment in the Payments list, click on the payment. Click on **Refund** button and the transaction will be marked as refunded.

10.3 Payment Receipt

You can download a payment receipt in the payment overview page by clicking on **Receipt** button.

10.4 Manually Creating a New Payment

To create a new payment, go to Invoices list page and select the invoice to pay. Click on **Pay** button and a number of fields will be displayed;

- **Amount**: The invoice amount will appear automatically by default. However, if the payment amount does not correspond to the default invoice amount, you can manually Enter the amount of payment received.
- **Payment Date**: The date the payment was received.
- **Payment Type**: Select the payment method that was used. Select the appropriate method from the list. Options include Bank Transfer, Cash, Razorpay, Stripe, PayPal, check and more.
- **Send Email**: Enable it to send a thank you email to customer.
- **Notes**: Here, you can add any comments or notes.

CHAPTER

ELEVEN

PROJECTS

If you're charging a customer for your work, you have the flexibility to charge fixed amounts as work is completed and/or set time-based billable rates. Hourly rates can be set for each team member.

11.1 Projects list

On the projects list page you'll see a table with the columns below;

- **Name**: Project title/name
- **Client**: The name of the client
- **Members**: Team members attached to the project
- **Amount**: The current total cost of the project
- **Cost**: Cost of the project so far
- **Billing Method**: Shows the billing method type

11.2 Create Project

To create a new project, go to the Projects tab on the main sidebar, and click on the **+New Project** button. This will open the project create modal offering a series of text and numerical inputs.

The form contains:

- **Name**: Project title/name
- **Customer**: Select the relevant client from the client list.
- **Start Date**: Project start date
- **Due Date**: Expected end date for the project
- **Billing Method**: There are four billing methods and each one is explained below;
- **Est. Hours**: Estimated number of hours for the project.
- **Description**: Summary of the project.

Tip: You can create a new client while creating a new project. Simply click on the **New Client** link, situated next to Customer field. A pop-up modal will open, enabling you to complete the new client's details. Then continue creating the project for this new client.

11.2.1 Billing Methods

Hourly Staff Rate

If you have Staff in your account, their rate is how much your Client will be billed regardless of the Task the hours are logged against. This method is really useful if you have multiple Staff working on a Project all with different rates.

Hourly Task Rate

Your Client will be billed based on the Tasks that are assigned to the Project. It doesn't matter who tracked the time, only the **Task Rate** is being taken into consideration. This method is really useful if you charge different rates for different Tasks regardless of who did the work.

Hourly Project Rate

It doesn't matter who tracked the time, or which Task was chosen, your Client will always be billed at one consistent rate for that Project.

Fixed Project Amount

It doesn't matter how many hours were logged, the Project will always be billed at the Flat Project Amount. You set the Flat Project Amount from the Edit Project screen.

11.3 Time Entries

11.3.1 Manual Entry

- To enter time manually click on **More** button and choose Timesheets.
- Click **New Entry** button
- Select a task from the dropdown list.
- Choose a start time and end time
- Enter time entry description and choose whether it's billable or unbillable.

Click **Save** when completed.

11.3.2 Start Timer

To start a timer.

- Open specific task and click **Actions** at the top bar of the task details slide panel. Click **Start Timer** to start timer for this task.
- Whenever there is a timer running, the left sidebar will show the number of running timers next to Projects link.
- You can stop the timer by following the same process of starting the timer above.

11.4 Invoice Project

In Sereni, you and your team can track your time to Projects. Those Projects are assigned to individual customers, and you can invoice your clients easily based on that tracked time by generating an Invoice with time entries on it.

To generate an invoice.

- Open the project dashboard click **Invoices** to start invoicing your project. Click **New Invoice** to begin.
- Select **Add billable time entries** if you want to invoice unbilled time entries.
- Select **Add billable expenses** if you want to add unbilled expenses to the invoice.

Click **Save** to generate project invoice.

Tip: Billed time entries will all be marked as **Billed**.

CHAPTER

TWELVE

TASKS

Projects are made up of tasks, and knowing how to manage your tasks (and everyone else's) is the secret to getting your projects completed on time. The ability to keep track of your tasks and deadlines makes the task management feature a crucial tool for your freelance business needs.

12.1 Tasks List

The Tasks list displays a list of all your tasks, for all projects, in an easy-to-follow table format.

Tip: You can mark a task as completed by simply clicking on the checkbox that appears on the left side of the task name.

12.2 Create Task

12.2.1 Task List

To create project task stages, open a project and click on tasks tab. Click on the **New List**. A modal will popup that allows you to add a new task list.

12.2.2 New Task

To create a new task, open the project you want to create a task for. Click on **tasks** tab and select **Add Task** button.

The task creation form contains the fields below;

- **Task Name**: Enter the task name
- **Milestone**: Select a milestone attached to this project (optional)
- **Board**: Enter the task stage e.g Backlog
- **Description**: Enter a task description
- **Start Date**: Date when the task should start
- **Due Date**: Date when the task should end
- **Hourly Rate**: The billing rate to charge per hour for the task
- **Est. Hours**: Total number of estimated hours a task takes

- **Recurring Task**: Select the frequency in which the task repeats.
- **Visible to Client**: Uncheck it to hide from client

12.3 Recurring Tasks

You can create recurring tasks for tasks you have to do again and again as part of a single project. Before setting a repeat, you will first need to make sure you have a due date set on the task.

When editing the task, you will see the **Recurring Task** field and under this field you can choose when the task should repeat:

You can set a repeat interval such as `daily, week, month, quarter, six months, year`.

Once you have chosen the interval, you can decide for the repeat to stop by a particular date.

Note: Any files attached to the original task will not carry over to each future repeating task.

Tip: To stop a recurring task, edit the task and change **Recurring Task** field to **No**.

Tip: Reminders are sent based on the due date of the task.

CHAPTER

THIRTEEN

EXPENSES

Track all of your lunch meetings or training materials, and other miscellaneous expenses in project expenses section, and instantly bill clients. Expenses allow you to capture the billable and non-billable costs that you incur as part of your project work. These non-labor expenses are usually itemized costs for things like materials, travel expenses, or fixed-fee services. Typically, billable expenses are passed on to a client or customer, whereas non-billable expenses are internal costs paid for by your employer.

Tip: You can create Expenses that you incur as part of your business operations (for example, Internet Expenses), or Recurring Expenses for those charges you incur on a frequent basis.

13.1 List Expenses

To view the Expenses list page, click on specific project and select expenses tab.

13.1.1 Overview

The Expenses list page displays a list of all project expenses that you choose to enter. The table columns appear as below;

- **Name**: Expense title
- **Category**: The assigned category of the expense
- **Amount**: The expense amount including taxes
- **Expense Date**: The date the expense occurred

13.2 Expense Categories

Sereni makes it easy to keep track of your spending with categories to organize them.

You can also create your own custom categories if a specific one isn't available.

13.2.1 Adding/Editing Categories

Go to **Settings > Expense Categories**. Here you can add, edit and delete your expenses categories.

13.3 Create Expense

You can create a new expense directly from project expenses list page by clicking on the **New Expense** button. A modal will open for you to enter expense information.

- **Amount**: The amount of the expense.
- **Category**: Select the category from the drop-down list.
- **Vendor**: Enter a vendor associated wit this expense
- **Notes**: Enter a description of the expense. When the expense is converted to an invoice, the text you enter here will feature as the line item description for the expense on the invoice.
- **Expense Date**: Date when the expense incurred
- **Recurring Expense**: Whether the expense should recur
- **Billable**: Whether the expense is billable
- **Receipts**: Upload expense receipts

13.4 Recurring Expenses

Some expenses are incurred on a consistent basis over a period of time, and manually recording them each time can get really tedious. Generating these expenses can be automated in Sereni, resulting in systematic tracking.

To make an expense recur, edit the expense and modify **Recurring Expense** field.

You can set a repeat interval such as `week, month, quarter, six months, year`.

Once you have chosen the interval, you can decide for the repeat to stop by a particular date.

Tip: To stop a recurring expense, edit the expense and change **Recurring Expense** field to **No**.

13.5 Invoice Expense

Are you billing a client directly for an expense? To bill an expense, first create an invoice for the client. Open the new invoice and just below the Client address, you'll see a button **Add Expenses** click on the button to open the modal where you can select the expenses to include in the invoice.

CHAPTER

FOURTEEN

REPORTS

You can only manage what you can measure. With Sereni, manage the way your business is run with reporting that breaks down key metrics relating to your sales trends, invoices, projects, expenses and more. Reports are a great way to not only see how you're doing, but also forecast where you're going in the future.

Attention: All the charts and statistics displayed in the reports section will use your base currency.

14.1 Invoice Report

Your invoice report gives you a comprehensive overview of all invoices you have generated in a given time period.

There are a few options you can change when generating reports;

- **Date Range** - You can set a date range you would like this report to include information within
- **Customer** - You can run Client(s) specific Invoice reports if necessary
- **Project** - Choose to only include project specific invoices
- **Status** - Choose to only include Invoices with specific statuses (such as Not Paid, Fully Paid, Cancelled etc)
- **Balance** - Only display invoices with balances

14.2 Expenses Report

The Expense Report breaks down all of your expenses into detail. You can run expenses report by doing the following:

- Click on the Reports tab
- Select Expenses from the top bar

There are a few options you can change when generating reports;

- **Date Range** - You can set a date range you would like this report to include information within
- **User** - Choose to show expenses created by specific user
- **Customer** - Choose to show expenses for a specific client
- **Project** - You can run reports for a specific project
- **Billable** - Choose to only include billable expenses
- **Invoiced** - Only display billed/unbilled expenses

- **Category** - Only display expenses in a specific category

14.3 Payments Report

Your Payments Report breaks out all of the payments you have recorded in Sereni. You can run payments report by doing the following:

- Click on the Reports tab
- Select Payments from the top bar.

There are a few options you can change when generating reports;

- **Date Range** - You can set a date range you would like this report to include information within
- **Customer** - Choose to show expenses for a specific client
- **Invoice** - Choose to only include payments for an invoice
- **Project** - You can run reports for a specific project
- **Payment Method** - Only display payments received in a specific payment provider

14.4 Timesheet Reports

The Timesheet Report is a very helpful way for your business to review how much time is being spent on a given project and by whom. Time Reports show the hours and billable information for each project, task and team member.

You can run time report by doing the following:

- Click on the Reports tab
- Select Timesheets from the top bar.

There are a few options you can change when generating reports;

- **Date Range** - You can set a date range you would like this report to include information within.
- **User** - Include only those time entries created by a particular user
- **Project** - Choose to show time entries for a specific project
- **Task** - Choose to show time entries for a specific task
- **Billable** - Include only those time entries that are billable or unbillable.

CHAPTER

FIFTEEN

SYSTEM SETTINGS

You can view and edit the system configuration in your portal by going to **Settings** and clicking on the **System Settings**.

You can modify the settings below;

- **Default Language**: The default system language
- **Locale**: Your preferred system locale
- **Timezone**: Your timezone
- **Date Format**: Choose your preferred date format
- **Privacy Policy URL**: A link to your privacy policy (Shown when registering)
- **Default Role**: When a user is created or registered, this role will be assigned.

> **Attention:** Do NOT set the **Default Role** to Admin.

Other Options

- **Enable ReCaptcha**: Use ReCaptcha in registrations
- **Login Color**: Login background color.
- **Rows per table**: Number of records to be displayed in each table.
- **Stop timer logout**: When this option is enabled, running timers will be stopped when you logout.

CHAPTER

SIXTEEN

INVOICE SETTINGS

You can view and edit the invoice configuration in your portal by going to **Settings** and clicking on the **Invoice Settings**.

Note: Any changes you make to the Invoice Settings will apply to all your invoices.

You can modify the settings below;

- **Invoices due after**: Number of days before an invoice is overdue.
- **Invoice Number Format**: Invoice format. Default -[yyyy][mm][dd]-[i] for {year}{month}{day}-{number}
- **Invoice Prefix**: Add your chosen prefix. For example, you may choose to add your company initials, such as AD and Invoices will appear as AD0001 etc Default **INV**
- **Reminder 1**: Number of days when a standard overdue reminder is set to send. Default 1
- **Reminder 2**: Number of days when a second reminder should be sent after invoice overdue. Default 5
- **Reminder 3**: Number of days when a late fee should be applied and final reminder sent. Default 10
- **Apply late fee on last reminder**: Apply late fee to invoice on 3rd reminder
- **Late Fee**: Late fee percentage
- **Send invoice on recur**: Automatically send invoice to customer when they recur
- **Invoice Logo**: Upload your Invoice Logo
- **Notes**: Any additional information that should appear at the bottom of your invoices

CHAPTER

SEVENTEEN

PAYMENT SETTINGS

You can view and edit the payment configuration in your portal by going to **Settings** and clicking on the **Payment Settings**.

Note: Supported payment gateways are **paypal, stripe, mollie, razorpay, bank, gocardless**

You can modify the settings as below;

- **Payment Prefix**: Add your chosen transactions prefix. Default PAY
- **Payment Number Format**: Estimate format. Default -[yyyy][mm][dd]-[i] for {year}{month}{day}-{number}
- **Enabled Payments**: Enable payment providers.
- **Bank Details**: Your Bank information to be displayed to client incase they need to pay via Bank

CHAPTER

EIGHTEEN

CONFIGURATION SETTINGS

You can view and edit configuration in your portal by going to **Settings** and clicking on the **Configuration Settings**.

Note: This settings modify your .env configuration file.

CHAPTER

NINETEEN

TRANSLATIONS

Translations are a collection of visual elements in Sereni application, like labels, information messages, notifications, alerts, statuses, etc

Translation strings may be defined within JSON files that are placed within the /lang directory. When taking this approach, each language supported by your portal would have a corresponding JSON file within this directory. You can follow laravel guide on how to add translation strings here

19.1 Adding Translations

If you want to add translations to Sereni;

- Create a JSON file for your language example `es.json` inside /lang folder.
- Copy contents of english file `en.json` and paste it to your new file `es.json`
- After copying you can now modify the json line values
- Lastly register your language in Settings > System Settings > Languages section. For language **code** enter **es** and give it a name e.g **spanish**