Document 02 – Applicatiestructuur en Mappenoverzicht

1. Hoofdstructuur van het project
```
Gemeente/
├── app/                → Domeinlogica (controllers, services, models)
├── bootstrap/          → Framework bootstrap en cache helpers
├── config/             → Applicatieconfiguratie (logging, queue, mail…)
├── database/           → Migrations, seeders, factories
├── public/             → Webroot (index.php, assets, logo's)
├── resources/          → Blade views, JS (Vite), CSS
├── routes/             → Web/api/console routes
├── storage/            → Logs, cache, uploads
├── tests/              → Feature & unit tests (Pest)
└── vite.config.js / package.json → Frontend toolchain
```

2. `app/` directory
2.1 `app/Http/Controllers`
- `Api/ChatController.php` beheert chatbot-endpoints.
- `Api/ComplaintApiController.php` levert CRUD/zoek endpoints voor klachten.
- `Admin`, `Auth`, `Web` submappen bevatten respectievelijk beheerportaal, authe
nticatie en publieke pagina's.

2.2 `app/Http/Middleware` & `Requests`
- Middleware zoals `Authenticate`, `VerifyCsrfToken` en maatwerk policies voor t
oegangscontrole.
- Form Requests (bijv. `ProfileUpdateRequest`) centraliseren validatie.

2.3 `app/Models`
- `Complaint`, `Attachment`, `Note`, `StatusHistory`, `Setting`, `User`.
- Modellen hebben `$fillable`, `$casts` en relaties (`Complaint::attachments()`,
 `::notes()`, `::statusHistories()`).

2.4 `app/Services`
- `ChatbotService` (intents + responses) en `PrivacyLogger` (PII-vrije logging).
- Eventuele extra services (bijv. notificaties) worden hier toegevoegd om contro
llers schoon te houden.

2.5 Overige mappen
- `Console/Commands` voor artisan-commando's.
- `Policies`, `Notifications`, `Events`, `Listeners`, `Jobs` ondersteunen uitgeb
reide domeinlogica.

3. `database/`
3.1 Migrations (`database/migrations`)
- `2025_09_22_091228_create_complaints_table.php` creëert de basisvelden voor kl
achten.
- `2025_09_22_134511_add_fields_to_complaints_table.php` breidt uit met categori
e, prioriteit, contact- en toewijzingsinfo.
- `create_attachments_table`, `create_notes_table`, `create_status_histories_tab
le` leggen relaties per klacht vast.
- `create_settings_table` en `create_afspraken_table` ondersteunen respectieveli
jk systeeminstellingen en afsprakenmodule.

3.2 Seeders & factories
- Placeholder-structuur aanwezig (`database/seeders`, `database/factories`) zoda
t demo-data eenvoudig kan worden toegevoegd.

4. `resources/`
4.1 `resources/js`
- `chatbot.js` bevat de volledige widget (UI + API-calls).

- Andere bundels (bijv. dashboard) worden via Vite samengevoegd in `resources/js/app.js`.

4.2 `resources/views`
- Blade-layouts (`layouts/app.blade.php`, `layouts/guest.blade.php`) en pagina's voor burgers/ambtenaren.
- Componenten (bijv. `components/application-logo.blade.php`) centraliseren herbruikbare UI.

4.3 `resources/css` of `resources/sass`
- Tailwind of custom styles; Vite bundelt dit richting `public/build`.

5. `public/`
- `index.php` is de front-controller die `bootstrap/app.php` laadt.
- Statische assets zoals `images/chatbot-logo-small.svg`, uploads, `robots.txt`.
- De output van Vite (`public/build/…`) wordt hier bediend.

6. `routes/`
- `web.php` voor server-side rendered pagina's.
- `api.php` voor JSON endpoints, inclusief chatbot (`/api/chat`), klachten (`/api/complaints`), statistieken en zoekfunctionaliteit.
- `console.php` registreert artisan-commando's en `channels.php` definieert broadcasting-kanalen.

7. `config/`
- Belangrijke bestanden:
  - `app.php`: app-naam, timezone, locale (`nl`).
  - `logging.php`: definieert `privacy_safe`, `security`, `audit` kanalen.
  - `queue.php`, `cache.php`, `database.php`: infrastructuurinstellingen.
- `.env` overschrijft runtime secrets (DB, mail, log level).

8. `storage/`
- `logs/` bevat `laravel.log`, `application.log`, `security.log`, `audit.log`.
- `app/public` is symlinkbaar naar `public/storage` voor uploads.
- `framework/cache`, `framework/sessions` voor runtime gegevens.

9. `tests/`
- Pest test suites onder `tests/Feature` en `tests/Unit`.
- Voorbeeld: `tests/Feature/SecurityTest.php` verifieert dat privacy logging correct werkt voor het chatbotkanaal.

10. Tooling en scripts
- `composer.json`: Laravel 12, PHP 8.2+, packages als `spatie/laravel-permission`, `intervention/image`.
- `package.json`: Vite 7, Tailwind 3, Alpine.js, axios, concurrently.
- `npm run dev` start Vite, Laravel serve en queue worker via `concurrently`.
- `composer dev` script start volledige ontwikkelstack incl. `php artisan serve`, `queue:listen`, `pail` logging en `npm run dev`.

Dit overzicht laat zien hoe elke map een duidelijke verantwoordelijkheid heeft, zodat uitbreidingen (nieuwe intent, extra module, UI-aanpassing) snel kunnen worden geplaatst op de juiste plek zonder technische schuld op te bouwen.