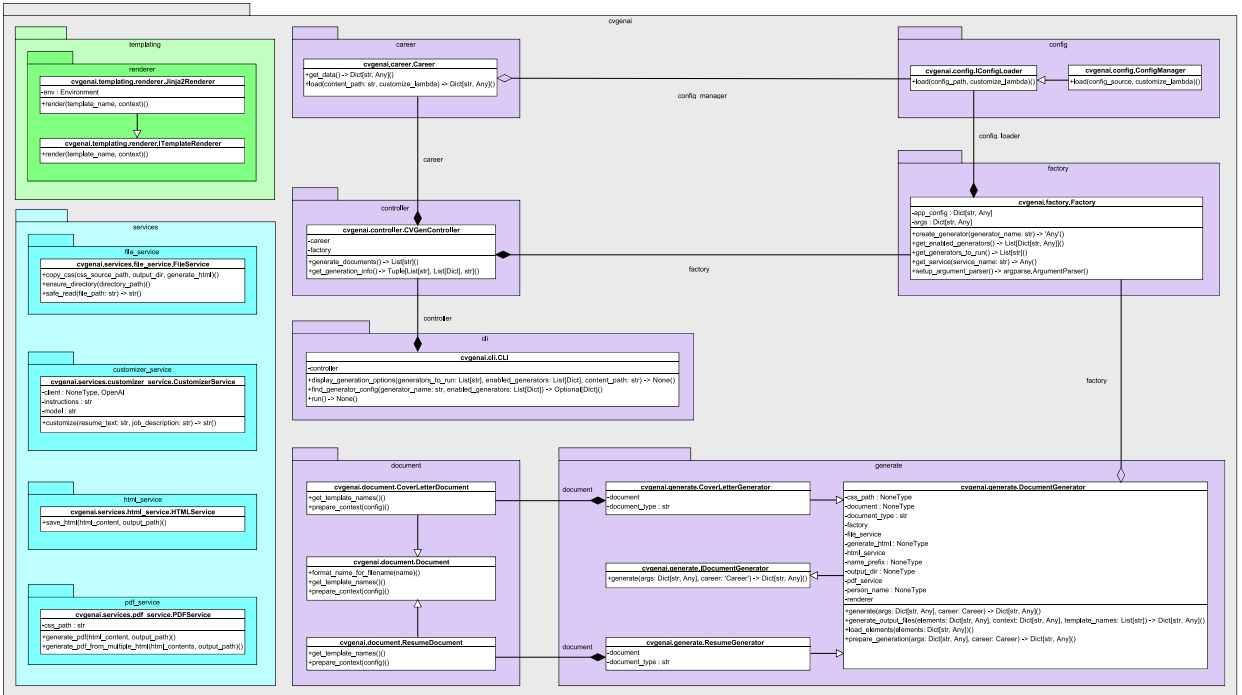
cvgenai
templating
renderer
cvgenai.templating.renderer.Jinja2Renderer
-env : Environment
+render(template_name, context)()
cvgenai.templating.renderer.ITemplateRenderer
+render(template_name, context)()
services
file_service
cvgenai.services.file_service.FileService
+copy_css(css_source_path, output_dir, generate_html)()
+ensure_directory(directory_path)()
+safe_read(file_path: str) -> str()
customizer_service
cvgenai.services.customizer_service.CustomizerService
-client : NoneType, OpenAI
-instructions : str
-model : str
+customize(resume_text: str, job_description: str) -> str()
html_service
cvgenai.services.html_service.HTMLService
+save_html(html_content, output_path)()
pdf_service
cvgenai.services.pdf_service.PDFService
-css_path : str
+generate_pdf(html_content, output_path)()
+generate_pdf_from_multiple_html(html_contents, output_path)()
career
cvgenai.career.Career
+get_data() -> Dict[str, Any]()
+load(content_path: str, customize_lambda) -> Dict[str, Any]()
config
cvgenai.config.IConfigLoader
+load(config_path, customize_lambda)()
cvgenai.config.ConfigManager
+load(config_source, customize_lambda)()
config_manager
config_loader
career
factory
controller
cvgenai.controller.CVGenController
-career
-factory
+generate_documents() -> List[str]()
+get_generation_info() -> Tuple[List[str], List[Dict], str]()
factory
cvgenai.factory.Factory
-app_config : Dict[str, Any]
-args : Dict[str, Any]
+create_generator(generator_name: str) -> 'Any'()
+get_enabled_generators() -> List[Dict[str, Any]]()
+get_generators_to_run() -> List[str]()
+get_service(service_name: str) -> Any()
+setup_argument_parser() -> argparse.ArgumentParser()
controller
factory
cli
cvgenai.cli.CLI
-controller
+display_generation_options(generators_to_run: List[str], enabled_generators: List[Dict], content_path: str) -> None()
+find_generator_config(generator_name: str, enabled_generators: List[Dict]) -> Optional[Dict]()
+run() -> None()
document
cvgenai.document.CoverLetterDocument
+get_template_names()()
+prepare_context(config)()
cvgenai.document.Document
+format_name_for_filename(name)()
+get_template_names()()
+prepare_context(config)()
cvgenai.document.ResumeDocument
+get_template_names()()
+prepare_context(config)()
document
document
generate
cvgenai.generate.CoverLetterGenerator
-document
-document_type : str
cvgenai.generate.IDocumentGenerator
+generate(args: Dict[str, Any], career: 'Career') -> Dict[str, Any]()
cvgenai.generate.ResumeGenerator
-document
-document_type : str
cvgenai.generate.DocumentGenerator
-css_path : NoneType
-document : NoneType
-document_type : str
-factory
-file_service
-generate_html : NoneType
-html_service
-name_prefix : NoneType
-output_dir : NoneType
-pdf_service
-person_name : NoneType
-renderer
+generate(args: Dict[str, Any], career: Career) -> Dict[str, Any]()
+generate_output_files(elements: Dict[str, Any], context: Dict[str, Any], template_names: List[str]) -> Dict[str, Any]()
+load_elements(elements: Dict[str, Any])()
+prepare_generation(args: Dict[str, Any], career: Career) -> Dict[str, Any]()