**P3 SOFTWARE ARCHITECTURE TEMPLATE**

O template P3 une o modelo de documentação de arquitetura 4+1 com o C4 em uma abordagem DDD e ágil, na forma de uma documentação não-abrangente e incremental. O template é organizado em 3 visões: produto, projeto e processo. Cada uma representando respectivamente a perspectiva do usuário, do desenvolvedor e do administrador (infraestrutura).

Apenas o nível mais abstrato é obrigatório na especificação dessa documentação, o detalhamento maior deve ocorrer conforme demanda concreta seguindo o princípio YAGNI e o Simple Design do XP. Uma maior simplicidade da documentação indica maior liberdade de escolha no desenvolvimento, um maior detalhamento comunica restrições ou requisitos que devem ser seguidos.

A abordagem dessa documentação é apresentar níveis diferentes de abstração (profundidade) na modelagem que possam avançar (detalhar) apenas quando necessários em seu contexto, evitando assim uma documentação abrangente, com design antecipado (upfront) ou detalhamento desnecessário/excessivo. A seguir é apresentado as perspectivas com suas camadas de abstração.

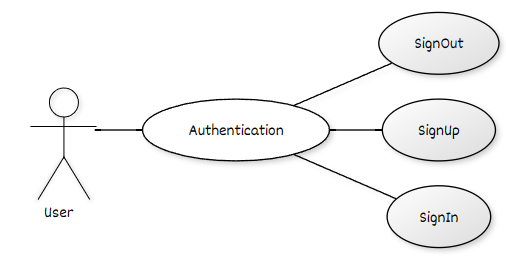
|  |  |  |
| --- | --- | --- |
| **PRODUCT** (user) | **PROJECT** (dev) | **PROCESS** (adm) |
| User diagram | Layer diagram | Deploy diagram |
| User Story specification | Domain model |  |
| Business Process diagram | Infrastructure list |  |

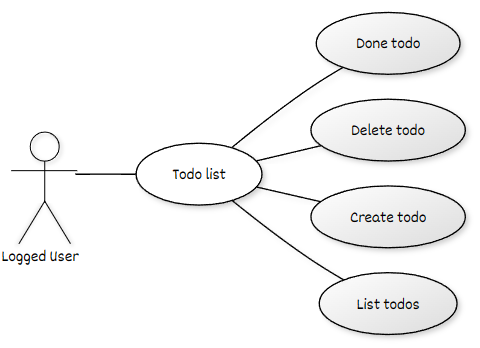
Esse template também sugere ferramentas simples, ágil e gratuita que como sugestão é utilizada em linguagens adaptadas do UML sugeridas aqui como user model, layer model, etc. Também são utilizadas outras ferramentas como o Bizagi para o processo de negócio.

**Product**

A visão de produto contém a perspectiva do usuário em 3 níveis de abstração a seguir, onde apenas o primeiro nível de abstração é obrigatório, os outros dependerão da demanda por maior detalhamento ou para informar um requisito ou restrição.

* **Diagrama de usuário**: um diagrama de caso de uso que representa um recurso (em branco) com seu ator e suas funcionalidades relacionadas.
* **História de usuário**: detalha uma funcionalidade usado o template de “Como ... Quero ... Para ...” descrevendo seus critérios de aceitação ao lado.
* **Diagrama de processo**: para cenários mais complexos ou mais críticos, fornece mais precisão sobre os detalhes de fluxo da história de usuário.





**Authentication**

|  |  |  |
| --- | --- | --- |
| **Feature** | As a … I want … So that I … | * Acceptance criteria 1 * Acceptance criteria 2 |

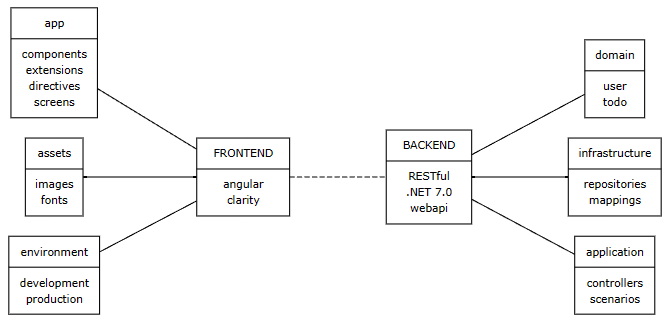
|  |  |  |
| --- | --- | --- |
| **SignIn** | As a new User I want signIn my user So that I enter in system | * The password field must be masked |

|  |  |  |
| --- | --- | --- |
| **SignUp** | As a new User I want signup my data So that I can enter in system | * Username will be the email * Must have name, birth and role * Password rule with numbers, lowercase and uppercase letter |
| C:\Users\Jonathan\AppData\Local\Microsoft\Windows\INetCache\Content.Word\signup.png | | |

**Project**

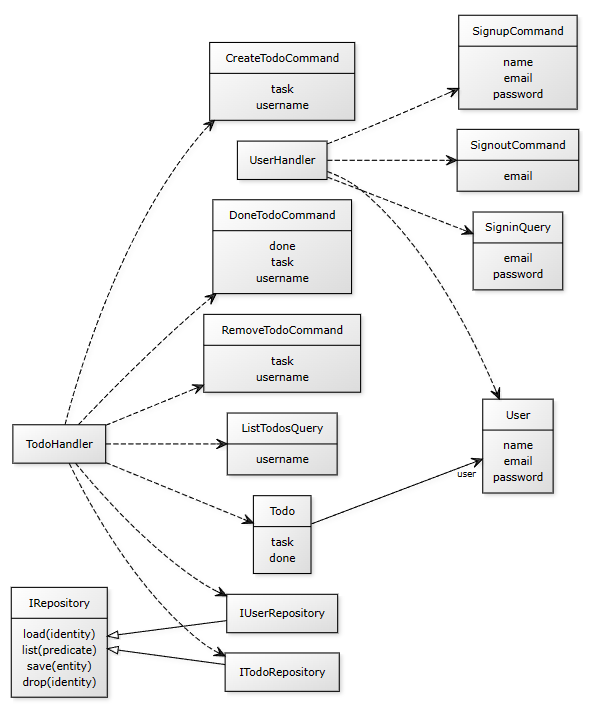
A visão de produto contém a perspectiva do usuário em 3 níveis de abstração a seguir, onde apenas o primeiro nível de abstração é obrigatório, os outros dependerão da demanda por maior detalhamento ou para informar um requisito ou restrição.

* **Esquema de projetos**: representa cada projeto (maiúsculo) do sistema contendo suas respectivas camadas (minúsculo) e módulos (estrutura).
* **Modelo de camadas**: modela uma camada de projeto. Os módulos contém o perfil de frontend ou backend. Os modelos de frontend são protótipos de telas, componentes e/ou estilos. Os modelos de backends se iniciam no API design, seguem pelo para o tech stack (infrastructure) e concluim no Domain Model.
* **Diagrama de classes**: para cenários mais complexos ou mais críticos, algumas classes podem ser definidas utilizando um diagrama de classes.

****

**Backend**

* POST /user/signup { name,email,password }
* GET /user/signin {token}
* DELETE /user/signup/email {token}
* POST /todo {token}
* DELETE /todo/:task {token}
* PUT /todo/:task {token}

****