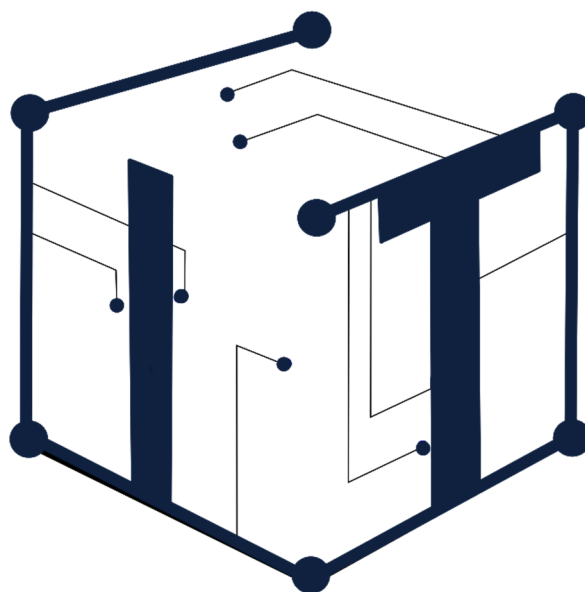


Projeto Final de Engenharia Informática

Relatório Final



inventiveTr@ining

Sistema de acompanhamento online de projetos de
microeletrónica

Aluno: Duarte Nuno Dutra Borges Cota

Número de aluno: 1600308

Email: 1600308@estudante.uab.pt ; duartenunocota@gmail.com

Contacto telefónico: +351 914 762 331

Orientador: Leonel Morgado

24 de junho de 2020

AGRADECIMENTOS

Ao Orientador do projeto, Doutor Leonel Morgado, pela confiança manifestada ao permitir a integração deste trabalho e do seu autor no projeto internacional de investigação Inven!RA, pela sua sábia orientação, pela sua total disponibilidade e pela clareza dos seus ensinamentos e explicações.

Ao colega Tiago Cruzeiro, membro da equipa de investigação Inven!RA, no âmbito do seu projeto de Mestrado na Faculdade de Engenharia da Universidade do Porto, de implementação da plataforma, um agradecimento pela disponibilidade e flexibilidade de articulação de trabalhos de desenvolvimento que evidenciou o longo do semestre.

Índice

1. OBJETIVOS, ENQUADRAMENTO E CONTEXTO INICIAL.....	5
1.1 Objetivos	5
1.2 Enquadramento.....	6
1.3 Contexto inicial	7
2. DESENVOLVIMENTO	8
2.1 Escolha do caso de estudo e análise preliminar.....	8
2.2 Identificação e parametrização de subatividades	9
2.3 Grafo de subatividades	10
2.4 Desenho e desenvolvimento de subatividades.....	11
2.5 Arquitetura do sistema.....	12
2.6 Implementação e utilização	13
3. IMPLEMENTAÇÃO DE OUTPUTS - ANALÍTICAS.....	23
4. TESTES DE INTEGRAÇÃO NA PLATAFORMA INVENIRA	27
5. TESTES COM UTILIZADORES	32
6. CRONOGRAMA.....	34
7. CONCLUSÕES	35
8. BIBLIOGRAFIA E RECURSOS WEB.....	38
9. ACEITAÇÃO DO ORIENTADOR.....	38
ANEXO 1 : Mapa de atividades e subatividades.....	39
ANEXO 2 : Códigos HTML frontend.....	41
ANEXO 3 : Códigos JavaScript frontend	49
ANEXO 4 : Códigos CSS frontend	65
ANEXO 5 : Códigos backend API Node JS	70
ANEXO 6 : Código-base + código teste hardware	89
ANEXO 7 : Aceitação do orientador.....	94

Índice de figuras

Figura 1 - Tabela de subatividades.....	10
Figura 2 - Grafo de subatividades.....	11
Figura 3 - Arquitetura do sistema – diagrama de componentes	12
Figura 4 - Estrutura tipo de ficheiro de registo	13
Figura 5 - Dashboard da ferramenta de autoria. Créditos: Tiago Cruzeiro – Inven!RA..	14
Figura 6 - Diagrama de sequência - configuração de atividades.....	15
Figura 7 - Página de configuração da atividade	15
Figura 8 - Configuração da atividade Consulta de Documentação - exemplo	16
Figura 9 - Diagrama de sequência - deploy atividade Consulta de documentação	17
Figura 10 - Deploy da atividade Consulta de Documentação.....	18
Figura 11 - Página da configuração da atividade de testes de firmware	19
Figura 12 - Página de configuração da atividade preenchida	19
Figura 13 - Deploy da atividade Testes de firmware	20
Figura 14 - Diagrama de sequência da atividade Testes de firmware	21
Figura 15 - Diagrama de sequência Process A	21
Figura 16 - Pacote JSON recebido no servidor (studentData).....	22
Figura 17 - MongoDB Atlas	22
Figura 18 - Objeto criado no servidor (Consulta de documentação).....	24
Figura 19 - Objeto criado no servidor (Configuração de hardware)	25
Figura 20 - JSON de resposta ao pedido de analíticas (Consulta de documentação)	26
Figura 21 - JSON de resposta ao pedido de analíticas (Configuração de hardware)	27
Figura 22 - Simulação de analíticas (Configuração de hardware).....	28
Figura 23 - Página de analíticas qualitativas (Configuração de Hardware).....	28
Figura 24 - Página de analíticas qualitativas –ver pacote de dados mais recente	29
Figura 25 - Página de analíticas qualitativas –ver todos.....	29
Figura 26 - Simulação de analíticas (Consulta de Documentação)	30
Figura 27 - Página de analíticas qualitativas (Consulta de Documentação).....	30
Figura 28 - UI de configuração de atividade. Créditos: Tiago Cruzeiro – Inven!RA	31
Figura 29 - UI de deploy de atividades. Créditos: Tiago Cruzeiro – Inven!RA.....	31
Figura 30 - UI consulta de analíticas. Créditos: Tiago Cruzeiro – Inven!RA	32
Figura 31 - Cronograma do projeto inventiveTr@ining.....	35

Acrónimos:

E@D	Ensino a Distância
LMS	<i>Learning Management System</i>
UFCD	Unidade de Formação de Curta Duração
ANQEP	Agência Nacional para a Qualificação e o Ensino Profissional
IAP	<i>Inventive Activities Plan</i>
JSON	JavaScript Object Notation
HTML	<i>Hypertext Markup Language</i>
CSS	<i>Cascading Style Sheet</i>
URL	<i>Uniform Resource Locator</i>
CET	Curso de Especialização Tecnológica
MCU	<i>Microcontroller Unit</i>
API	<i>Application Programming Interface</i>
REST	<i>Representational State Transfer</i>
UI	<i>User Interface</i>

1. OBJETIVOS, ENQUADRAMENTO E CONTEXTO INICIAL

1.1 OBJETIVOS

O objetivo do projeto *inventiveTr@ining* é criar uma ferramenta destinada a formadores, que lhes permita conceber, planear e acompanhar o desenvolvimento de projetos tecnológicos na área de Eletrónica e Automação. Consiste este projeto no desenho e desenvolvimento de componentes para permitir que uma plataforma online possa ser utilizada para acompanhamento de projetos curriculares no âmbito das UFCD de cursos profissionais (nível 4) e CET (nível 5), inseridas na área de formação 523-Eletrónica e Automação, de acordo com o Catálogo Nacional de Qualificações.

A plataforma de integração designa-se por *Inven!RA* (pronuncia-se “invenira”) e está em fase de desenvolvimento no Laboratório Associado INESC TEC, , em articulação com a UNISINOS (Projeto CAPES/PRINT “Transformação Digital e Humanidades”). Permite definir e acompanhar planos letivos, de forma integrada com sistemas LMS como o Moodle, a partir de módulos externos, programados à medida de cada atividade específica. Este projeto desenvolveu módulos para o objetivo acima enunciado, tendo decorrido em estreita articulação com o desenvolvimento da *Inven!RA*, que foi evoluindo e transformando-se durante o semestre, como é natural em atividades de investigação tecnológica.

Os módulos desenvolvidos, utilizados através da plataforma *Inven!RA*, permitirão aos docentes do ensino profissional apoiar, acompanhar e avaliar de forma mais estruturada os formandos/alunos ao longo do desenvolvimento deste tipo de atividades, que podem ser desenvolvidas em ambiente escolar ou extraescolar. Seguindo a terminologia da plataforma *Inven!RA*, os planos construídos com estas atividades designam-se por IAP (*Inventive Activities Plan* ou plano de atividades inventivas). Consistem num grafo linear ou não linear de instâncias de atividades parametrizáveis, associadas a objetivos de aprendizagem específicos por meio de uma combinação ponderada de *analytics*. A programação de alguns destes módulos de atividades, no contexto das UFCD, é o foco deste projeto. De notar que um IAP pode ser criado com atividades de diferentes *Activity Providers*, que podem ser empresas ou outras entidades dedicadas ao desenvolvimento de tecnologias educativas. Esta é uma das características distintivas da *Inven!RA*. Desta forma, as atividades desenvolvidas neste projeto podem ser concebidas como serviço terceiro (*third party*) para a *Inven!RA*, pelo que o projeto decorreu num contexto de integração de sistemas.

O formador assume em todo o processo dois papéis. No primeiro papel, na fase de conceção, age como *Learning Designer* (conceitor da aprendizagem) para conceber e criar na *Inven!RA* um plano para as atividades de projeto. No segundo papel, na fase de acompanhamento, age como formador, para atribuir o IAP aos seus formandos, executá-lo, acompanhar a evolução dos formandos e intervir pedagogicamente em conformidade.

Nesta fase de acompanhamento, o formador associa um IAP a um grupo de formandos (turma) e dá início à sua execução, quer tecnologicamente (ativando-o na plataforma) quer pedagogicamente (explicitando aos formandos a necessidade de iniciarem as

atividades). Os formandos acedem à plataforma LMS e iniciam as atividades preconizadas pelo formador, que utiliza as várias plataformas para a monitorização e ação pedagógica. Durante esta fase de execução do trabalho o formador:

- acompanha online, de forma síncrona ou assíncrona a evolução do trabalho dos formandos;
- recolhe informações sobre o estado de desenvolvimento dos projetos e dos resultados atingidos;
- presta esclarecimentos/apoio pedagógico aos formandos.

Em resumo contaremos com cinco intervenientes em todo o processo:

- Activity Provider: empresa/organização terceira (*third party*) que se dedica à criação de componentes de software (atividades) compatíveis com a plataforma Inven!RA. Para este projeto idealizou-se a empresa fictícia Duarte, Lda.;
- Inven!RA: plataforma de criação e acompanhamento de planos de atividades inventivas (IAP);
- Formador: profissional do ensino que age inicialmente como *learning designer* quando cria o IAP na Inven!RA, com as atividades disponibilizadas pela Duarte Lda. ou qualquer outro *Activity Provider*. Numa segunda fase gere o IAP, acompanha e orienta os formandos durante a execução do mesmo.
- Formando: executa o IAP realizando as atividades lançadas pelo formador no LMS e previamente configuradas na Inven!RA.
- LMS: plataforma de E@D em utilização na instituição de ensino. Neste trabalho é considerada a plataforma Moodle.

1.2 ENQUADRAMENTO

O E@D, não sendo propriamente uma metodologia recente, ganhou uma expressão talvez nunca vista em Portugal e no mundo graças à pandemia do novo coronavírus. Todos estes conceitos e designações associadas a este contexto entraram, de um dia para o outro, no nosso vocabulário quotidiano e rapidamente se percebeu que, apesar de não serem conceitos novos, a realidade mostra que o atual sistema de ensino em Portugal não estava preparado para uma mudança tão radical e imediata do tradicional ensino presencial para um ensino online com recurso a ferramentas digitais de comunicação e disponibilização de recursos de aprendizagem em plataformas LMS.

A emergência da situação acabou por precipitar uma mudança que há muito se exigia e sabia ser necessário, que se traduz na constatação do esgotamento do atual modelo de ensino português, e da necessidade de reinventar o sistema de ensino e aprendizagem potenciando a integração de ferramentas digitais atrativas, criativas e produtivas tirando partido na ampla predisposição das novas gerações para a utilização de equipamentos baseados em tecnologias de informação e comunicação.

Independentemente desta emergência ter revelado todas estas fragilidades e da certeza de que o ensino/formação não voltará a ser como no período pré-pandemia, a realidade é que muitos profissionais do ensino e formação já ensaiavam tentativas de mudança com a integração de metodologias digitais na planificação das suas atividades letivas e formativas. No ensino profissional, cuja realidade conheço bem e que deriva de mais de 17000 horas de formação lecionada em 25 anos de experiência, a estrutura curricular baseia-se em unidades de formação de curta duração (UFCD), tipicamente de 25 horas,

definidas no Catálogo Nacional das Qualificações da responsabilidade da ANQEP e divididas em três componentes – sociocultural, científica e técnica. Os formadores das UFCD técnicas lidam ano após ano com a insuficiência desta carga horária para, por um lado, cumprir os objetivos formativos definidos para cada UFCD; e por outro lado dar resposta às expectativas dos formandos que ingressam neste sistema de ensino, expectativa esta que assenta no desejo de ter uma formação maioritariamente em contexto *hands-on*, quer em laboratório quer em oficina. O projeto inventiveTr@ining surge como uma ferramenta capaz de introduzir uma alternativa simples, integrada e funcional para a gestão de projetos de alunos do ensino profissional/tecnológico em contexto “fora da escola” permitindo a definição de planos de atividades inventivas à medida dos objetivos do professor/formador com o necessário acompanhamento do progresso dos estudantes, tudo isto integrado numa plataforma única de arquitetura aberta.

1.3 CONTEXTO INICIAL

O projeto inventiveTr@ining nasceu da conjugação de dois fatores:

1. A intenção do autor de propor um trabalho, no âmbito da UC Projeto de Engenharia, com possível enquadramento na sua atividade profissional;
2. A oportunidade de integrar este trabalho no projeto de investigação Inven!RA, nomeadamente na fase de arranque da implementação da sua arquitetura, a decorrer no Laboratório Associado INESC TEC.

Importa salientar e descrever o estado inicial da plataforma Inven!RA para melhor enquadrar e justificar todo o trabalho desenvolvido neste projeto.

Sendo essencialmente uma plataforma de integração de serviços *Web*, pese embora a sua complexidade de arquitetura, a Inven!RA depende de serviços providenciados por entidades externas - *Activity Providers*. Neste âmbito o contexto inicial apontava para um quadro bastante primitivo no que toca a estes serviços externos:

- Inexistência de exemplos de serviços já desenvolvidos para a Inven!RA;
- Inexistência de um formato de dados padronizado para comunicação entre a Inven!RA e os *Activity Providers*;
- Inexistência de diagramas de fluxo de dados ou diagramas de arquitetura;
- Inexistência de formulários para configuração e *deploy* de atividades;
- Inexistência de exemplos para processamento de analíticas;

Este cenário implicou que o esforço de desenvolvimento se concentrasse, não só no trabalho de análise aplicável ao contexto de escolha do tema/ideia inicial, mas essencialmente na implementação de raiz, dos componentes de *software* de integração na plataforma Inven!RA, sem qualquer base de referência ou de adaptação, o que confere a este trabalho algumas componentes de dimensão inovadora e precursora.

Todo os componentes da API inventiveTr@ining foram desenvolvidos em articulação com a equipa do INESC TEC, responsável pela plataforma Inven!RA, exigindo um esforço de trabalho em equipa assinalável por forma obter-se uma prova de conceito funcional e integrada, que funcionará como referência para trabalhos futuros. Neste esforço foram, também, determinantes os parceiros internacionais do projeto Inven!RA que conferem ao projeto uma notável dimensão em conhecimento científico e tecnológico.

2. DESENVOLVIMENTO

Como já foi descrito, o objetivo fundamental deste projeto é criar um serviço intitulado *inventiveTr@ining* que oferece aos profissionais de docência componentes de software que lhes permite, quando recorrem à plataforma *Inven!RA*, dispor de atividades para estruturarem uma atividade de projeto inventivo (IAP) na área da Eletrónica e Automação.

2.1 ESCOLHA DO CASO DE ESTUDO E ANÁLISE PRELIMINAR

Para efeitos de desenvolvimento do projeto o primeiro passo foi identificar uma UFCD para efeitos de caso de estudo e estruturação de atividades na *inventiveTr@ining*. A escolha recaiu na UFCD 6073 – *Microcontroladores: aplicações* cujos objetivos e conteúdos estão definidos no Catálogo Nacional das Qualificações:

Objetivos:

- Controlar um *display* de cristais líquidos, através do programa do microcontrolador.
- Elaborar circuitos e programas adequados para controlar motores passo-a-passo.
- Implementar sistemas de aquisição de dados e controlo digital.
- Elaborar programas para controlo da velocidade de motores de corrente contínua por PWM.
- Reconhecer a estrutura de sistemas baseados em microcontroladores.
- Definir e aplicar funções relativas a endereços, dados e controlo.
- Desenhar fluxogramas.
- Programar microprocessadores/microcontroladores.
- Aplicar os microcontroladores no controlo de processos industriais.
- Identificar as principais funcionalidades do software de simulação e programação do microcontrolador em estudo.
- Programar e simular, em ambiente informático, o microcontrolador em estudo.
- Utilizar as principais características do microcontrolador.
- Interligar o microcontrolador com periféricos externos.
- Realizar hardware específico do projeto.
- Projetar o trabalho a desenvolver.
-

Conteúdos:

- Aquisição/tratamento de dados
- Controlo de temperatura
- Controlo de motores de corrente contínua (motores passo-a-passo, servos, PWM)
- Visualização de dados
- *Software* de simulação e programação (compilação e execução de programas)
- Portas paralelas
- Interrupções
- *Hardware* periférico

- Portas paralelas
- Interrupções
- Comunicação com periféricos/protocolos de comunicação
- Testes de *hardware* em placa de ensaio
- Realização de projeto aplicativo de controlo por microcontrolador
- Ensaio do projeto
- Relatórios intermédios e finais do projeto
- Memória descritiva, orçamento

Trata-se de uma UFCD que se enquadra perfeitamente no âmbito da prova de conceito que se pretende para este projeto, o que releva a utilidade e aplicabilidade de alternativas às atividades práticas em contexto de sala de aula que a inventiveTr@ining, em integração com a Inven!RA, oferece aos docentes e formadores.

Do processo de análise resultou a listagem de atividades de projeto que se segue e que engloba os objetivos acima identificados:

- Leitura do descritivo técnico da atividade;
- Interpretação do objetivo do projeto (resultado pretendido);
- Elaboração e preenchimento de mapa de quantidades (recursos materiais);
- Desenho de *hardware* (diagrama de ligações);
- Desenho de fluxogramas;
- Integração de *hardware*;
- Programação de componentes de *software*;
- Configuração de *hardware*;
- Comissionamento e testes;
- Elaboração de relatório.

2.2 IDENTIFICAÇÃO E PARAMETRIZAÇÃO DE SUBATIVIDADES

Cada atividade identificada no ponto anterior engloba diferentes objetivos e tarefas a cumprir pelos formandos. Para permitir uma melhor planificação cada atividade foi decomposta num conjunto de subatividades que permitem detalhar e sequenciar o desenvolvimento do projeto do formando, no âmbito da UFCD. Esta decomposição permite, por outro lado, identificar o conteúdo e estrutura dos formulários *Web*, específicos para cada atividade, que serão fornecidos aos docentes para criação dos IAP na Inven!RA, e definir as subatividades com precedência cuja transição dependerá do cumprimento de tarefas prévias (por exemplo, o formando deverá entregar um fluxograma do *firmware* do microcontrolador antes de iniciar a programação do mesmo). A tabela de subatividades resultante é a que se encontra na Figura 1.

Outra análise importante é a parametrização das subatividades, isto é, definição do tempo disponível para a realização das mesmas bem como os dados relevantes para que os formadores possam acompanhar os formandos durante o percurso destes no IAP. Desta avaliação resultou a tabela final que poderá ser consultada no Anexo 1 deste documento. Pode observar-se nesta tabela que a cada subatividade está associada uma limitação de tempo para a sua execução bem como os dados que o formador deverá dispor para acompanhar e orientar os formandos.

No.	Atividade	Subatividade	Descrição da atividade
1	A1	A1.1	ANÁLISE DE DOCUMENTAÇÃO Analisar o descritivo técnico da atividade. QUIZ_1: Questionário sobre a análise do descritivo técnico da atividade.
2	A2	A2.1	PREENCHIMENTO DE QUADRO QUIZ_2: Questionário sobre a estrutura e detalhe de um mapa de quantidades.
3		A2.2	PREENCHIMENTO DE QUADRO Elaborar mapa de quantidades dos recursos identificados como necessários. Submeter o mapa de quantidades apurado.
4	A3	A3.1	DESENHO DE HARDWARE QUIZ_3: Questionário sobre diagramas de ligações em eletrônica.
5		A3.2	DESENHO DE HARDWARE Desenhar o mapa de ligações (<i>hardware</i>) em <i>software</i> específico. Submeter o diagrama.
6	A4	A4.1	DESENHO DE FLUXOGRAMA QUIZ_4: Questionário sobre estrutura e conceção de fluxogramas.
7		A4.2	DESENHO DE FLUXOGRAMA Desenhar um fluxograma do algoritmo a implementar. Submeter o fluxograma.
8	A5	A5.1	INTEGRAÇÃO DE HARDWARE Integrar o hardware de acordo com diagrama desenvolvido em T3
9	A6	A6.1	PROGRAMAÇÃO DE COMPONENTES DE SOFTWARE QUIZ_5: Questionário sobre bases de desenvolvimento de software.
10		A6.2	PROGRAMAÇÃO DE COMPONENTES DE SOFTWARE Desenvolver o código para o microcontrolador (baseado em C++) Submeter o código.
11	A7	A7.1	CONFIGURAÇÃO DE HARDWARE QUIZ_6: Questionário sobre tipos de testes de funcionamento aplicáveis ao contexto e sobre o tipo de dados que deverão estar a ser gerados. Definir lista de testes a realizar.
12		A7.2	CONFIGURAÇÃO DE HARDWARE Fazer o <i>upload</i> do <i>firmware</i> ; <i>iniciar</i> testes de comissionamento e operabilidade.
13		A7.3	CONFIGURAÇÃO DE HARDWARE Refletir sobre o comportamento do protótipo (<i>hardware e software</i>) e melhorias.
14	A8	A8.1	ELABORAÇÃO DE RELATÓRIO Iniciar uma fase reflexão/autoavaliação.
15	A9	A9.1	ELABORAÇÃO DE RELATÓRIO QUIZ_7: Questionário sobre estrutura e conteúdo do relatório.
16		A9.2	ELABORAÇÃO DE RELATÓRIO Elaborar o relatório final. Submeter o relatório final.

Figura 1 - Tabela de subatividades

2.3 GRAFO DE SUBATIVIDADES

Para melhor compreender o fluxo de desenvolvimento do IAP elaborou-se um grafo de subatividades. Este grafo – Figura 2 – é determinante para a fase seguinte por duas razões.

1. Permite definir melhor o fluxo de transição entre subatividades;
2. Permite identificar subatividades com precedência e subatividades que podem decorrer em paralelo – ordem de realização pode ser uma opção do formando.

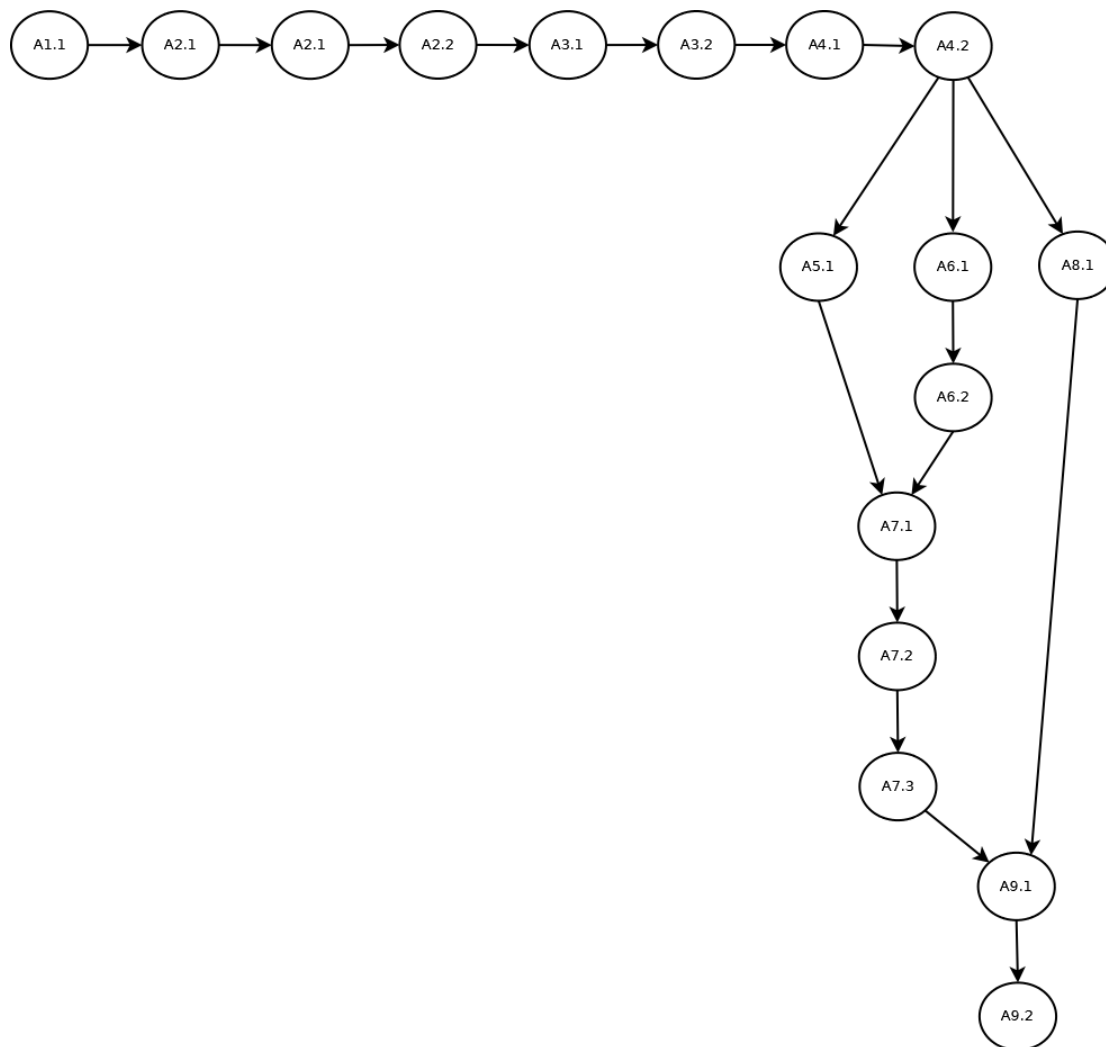


Figura 2 - Grafo de subatividades

As subatividades A1.1 a A4.2 possuem precedências pelo que se pode observar a sua serialização. As subatividades A5.1, A6.2 e A8.1 apenas dependem da conclusão da subatividade A4.2. Em termos concretos, assume-se que o formando poderá optar por qual atividade irá seguir: se para a atividades de integração de hardware (A5.1), programação de software (A6.1) ou começar a redigir o relatório (A8.1).

2.4 DESENHO E DESENVOLVIMENTO DE SUBATIVIDADES

De acordo com a filosofia de desenvolvimento do IAP na Inven!RA, todas as atividades são fornecidas pelo *Activity Provider*, no caso a empresa fictícia Duarte Lda. Tratando-se de um projeto enquadrado numa unidade curricular semestral do 3.º ano do curso de Licenciatura em Engenharia Informática, que decorre em paralelo com outras unidades curriculares, houve necessidade de definir as subatividades que seriam alvo de trabalho de desenvolvimento mais aprofundado (codificação e integração) sendo, à partida, assumido como inexequível, no período temporal e carga de esforço preconizados, o desenvolvimento de todas as subatividades. Optou-se por desenvolver as subatividades A1.1 e A7.2 como prova de conceito do sistema de integração da iventiveTr@ining na Inven!RA. Esta escolha adveio das suas características complementares enquanto provas de conceito. A atividade A1.1 é conceptualmente simples: uma consulta de documentação, exequível inteiramente numa interface *Web*. A atividade A7.2 é multiplataforma, requerendo a combinação de atos realizados numa interface *Web* com

atos efetuados autonomamente pelo aluno/formando num componente de *hardware* (Arduino). Esta complementaridade permitia ter uma prototipagem rápida das funcionalidades (atividade A1.1) e depois enriquecer a compreensão obtida nessa prototipagem com o contexto diferenciado da atividade A7.2.

2.5 ARQUITETURA DO SISTEMA

A compreensão do encadeamento dos processos de integração das diversas plataformas em jogo neste caso de estudo, bem como o papel e interação dos diferentes intervenientes com o sistema, é fundamental para perceber as opções tomadas quer a nível de configuração quer ao nível do desenvolvimento. Esta interação pode ser observada no diagrama de componentes da Figura 3 onde se representa a integração e interação entre os vários componentes do sistema:

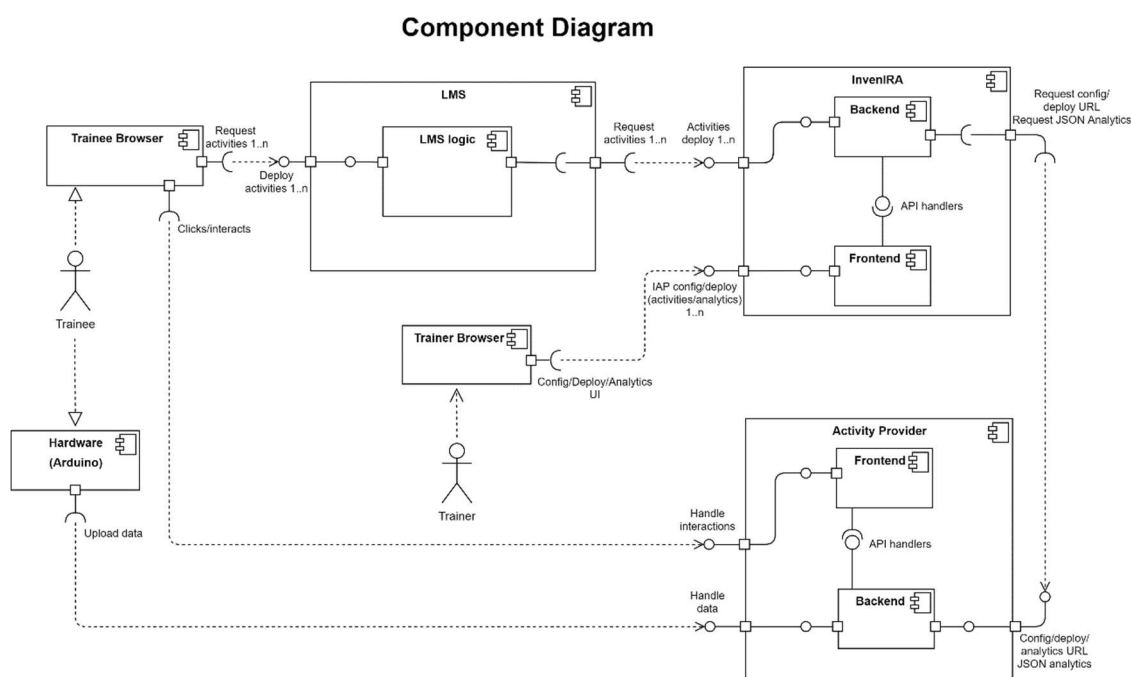


Figura 3 - Arquitetura do sistema – diagrama de componentes

Nesta imagem é claro o papel do *Activity Provider* como *third-party* em todo o sistema, responsável por gerir todos os serviços *Web* disponíveis no projeto InvenienteTr@ining. Para além de alojar as páginas *HTML* de configuração, o servidor do *Activity Provider* responde aos pedidos *HTTP* provenientes da plataforma Inven!RA, para todas as atividades de configuração e *deploy*. O servidor possui ainda os *handlers* configurados para lidar com as interações dos formandos com os formulários, para efeitos de rastreamento de análíticas, bem como para lidar com pedidos do *hardware* do aluno para armazenamento de dados e código.

No âmbito dos serviços disponibilizados pela Duarte Lda. existe um serviço de fornecimento de análíticas que é ativado a pedido da Inven!RA, para todas as atividades do *provider* configuradas num *IAP*, e que consiste num ficheiro *JSON* parametrizado de acordo com as informações de registo de cada atividade, para processamento pela plataforma integradora.

A arquitetura específica do projeto inventiveTr@ining baseia-se num modelo simples de arquitetura cliente-servidor. Foi desenhado e implementado um *Frontend* com uma *API* em *NODE JS*, responsável por gerir todos os serviços *Web* do sistema.

Backend (servidor HTTP): configuração dos *handlers* que gerem toda a lógica do sistema, ligação à base de dados e plataforma Inven!RA.

Frontend: páginas de UI para configuração de atividades, *deploy* de atividades e interação dos formandos –desenvolvidas em HTML, JavaScript, CSS. Consiste ainda no código implementado pelo aluno que depois de carregado no *hardware* irá gerar dados enviados automaticamente para o *Activity Provider*.

2.6 IMPLEMENTAÇÃO E UTILIZAÇÃO

A seguir faz-se a descrição da cronologia das diferentes fases que compõe o desenvolvimento e implementação da inventiveTr@ining e da sua operabilidade dentro do sistema gerido pela Inven!RA. Far-se-á uma simulação tendo em conta as diferentes fases, detalhando os processos e decisões tomadas em cada fase para que se possa compreender a totalidade da arquitetura do sistema.

Consideremos a empresa fictícia Duarte Lda. É um *Activity Provider* e dedica-se à criação de atividades inventivas específicas para o ensino profissional, nomeadamente para as UFCD de cursos da área de formação 523-Eletrónica e Automação. A Duarte Lda. decide criar atividades para integrar na plataforma Inven!RA, no âmbito da UFCD 6073 – Microcontroladores: aplicações. Para começar é desenvolvido o trabalho de análise da UFCD, decomposição de atividades em subatividades e conceção do grafo de subatividades, conforme descrito nas secções 2.1 a 2.3.

Terminado este processo, a Duarte Lda. regista na Inven!RA as suas atividades. Este processo ainda não está automatizado na plataforma Inven!RA, consistindo apenas no fornecimento manual à Inven!RA de um ficheiro JSON com campos definidos pela própria plataforma. Estes campos são fundamentais para que a Inven!RA saiba que dados fornecerá a Duarte Lda. na fase de *deploy* das atividades.

Na Figura 4 podemos observar um ficheiro-tipo de registo:

```
1 {  
2   "name": "activity name",  
3   "properties": [{  
4     "aprop": "smt"  
5   }],  
6   "config_url": "someconfigurl.html",  
7   "json_params": "someconfigurl.json",  
8   "style_url": "anurl.css",  
9   "analytics": []  
10 }
```

Figura 4 - Estrutura tipo de ficheiro de registo

Neste ficheiro ficam definidos os parâmetros que a Duarte Lda. se compromete a fornecer à Inven!RA no âmbito da integração de cada atividade nos IAP da plataforma

Após o processo de registo cada atividade fica instanciada na base de dados da Inven!RA, sendo-lhe atribuído um ID próprio.

Quando o formador acede à ferramenta de autoria (*Authoring Tool*) da plataforma Inven!RA, visualiza as atividades que estão disponíveis. Na Figura 5 observa-se o

dashboard da ferramenta de autoria da Inven!RA atualmente em desenvolvimento no INESC TEC:

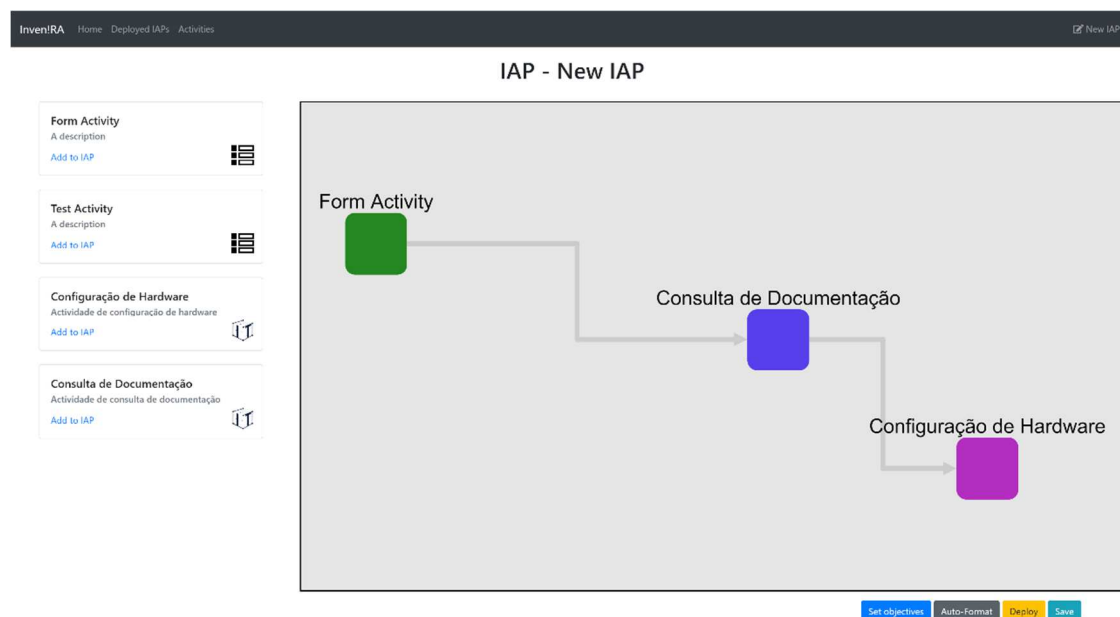


Figura 5 - Dashboard da ferramenta de autoria. Créditos: Tiago Cruzeiro – Inven!RA

O formador (no papel de *learning designer* ou concetor de planos de atividades) adiciona as atividades que pretende na área de trabalho. Na imagem, cada quadrado colorido corresponde a uma atividade. As atividades são depois ligadas entre si de acordo com a sequencialidade que o formador pretender.

Concluído este processo de inserção/ligação entre atividades, o formador procede à configuração das atividades (duplo clique sobre a atividade). A Inven!RA acede à informação de registo da atividade introduzida pela Duarte Lda. e mostra ao formador o formulário de criação da atividade alojado no *web site* da Duarte Lda.

De seguida descreve-se o processo de desenvolvimento e utilização das atividades escolhidas para demonstração de conceito, no âmbito deste projeto. São as atividades A1.1 – Consulta de documentação e A7.2 – Configuração de hardware - testes de *firmware*. Para cada atividade há que considerar, em qualquer circunstância, que o *Activity Provider* terá de disponibilizar um dispositivo para o formador configurar a atividade e outro dispositivo para o formando aceder à atividade e saber o que terá de desenvolver ou executar.

Configurar atividade: atividade de consulta de documentação.

Esta foi uma das atividades desenvolvidas no âmbito deste projeto e corresponde à subatividade A1.1 do grafo da Figura 2. Ao aceder à configuração desta atividade no IAP, a Inven!RA mostra ao formador a respetiva página do *Activity Provider* (a Duarte Lda.) que disponibiliza o formulário de introdução de documentação. Após o preenchimento do formulário os dados ficarão guardados na base de dados da Inven!RA, nos campos definidos no momento do registo da atividade com um ID próprio.

Na Figura 6 pode observar-se o diagrama de sequência do processo de criação da atividade consulta de documentação, este processo deverá ser, à partida, semelhante para qualquer atividade:

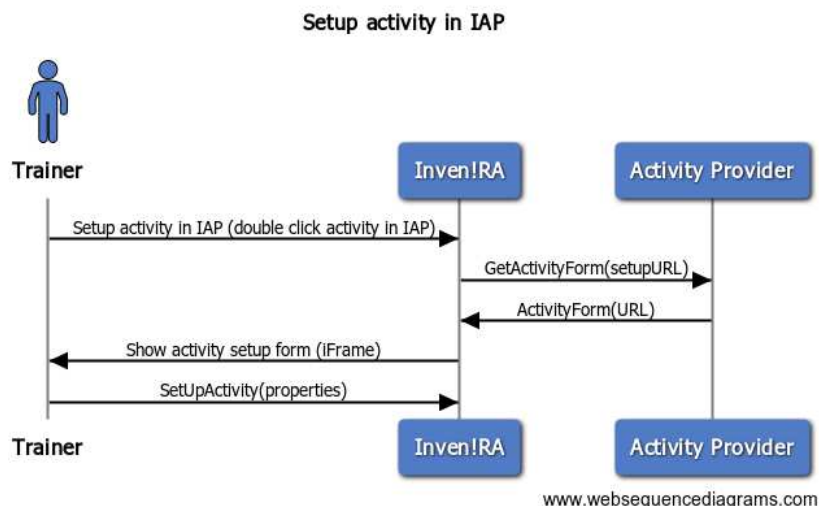


Figura 6 - Diagrama de sequência - configuração de atividades

Como já foi referido, a Duarte Lda., na qualidade de *Activity Provider*, é quem disponibiliza as páginas para configuração e posterior *deploy* das atividades. Relativamente à atividade de consulta de documentação o formulário desenvolvido é o da Figura 7. Este formulário é apresentado numa *iframe* HTML pela Inven!RA. Foi desenvolvido em HTML, CSS e JavaScript e os respetivos códigos encontram-se nos Anexos 2, 3 e 4 deste relatório. O formulário possui campos de preenchimento para:

1. introdução de um pequeno resumo do descritivo técnico do projeto que o aluno terá de desenvolver;
2. introdução de um URL para o descritivo técnico que o formador previamente carrega num qualquer serviço de *cloud hosting* tipo Google Drive ou Dropbox;
3. campos para adicionar URL para aceder a *datasheets* de componentes que o formando poderá utilizar no projeto, bibliografia, documentação técnica, etc.

Figura 7 - Página de configuração da atividade

Na Figura 8 observamos o resultado de uma simulação de configuração de uma atividade de consulta de documentação:

inventiveTr@ining
CRIAR CONSULTA DE DOCUMENTAÇÃO

1. **Resumo do descritivo técnico:**

Nesta actividade pretende-se planificar, desenhar, configurar e programar uma aplicação com micro-controladores para controlo de parâmetros ambientais numa estufa. Nesta página poderá descarregar o descritivo técnico do projecto bem como alguns documentos de apoio técnico e orientação. Os componentes indicados são meramente sugestões podendo ser utilizados outros desde que forneçam os dados no formato pretendido

Limpar

2. **URL para o descritivo técnico**

https://www.dropbox.com/s/w9k5ck7q4hcd39f/desc_tec1.doc?dl=0

3. **URL das hiperligações para materiais de apoio**

URL + Descritivo do URL:

Adicionar

Lista de hiperligações:

- ☐ <https://www.arduino.cc/reference/en/> -> Página de referência do Arduino
- ☐ <http://www.ti.com/lit/ds/symlink/lm35.pdf> -> Datasheet LM35
- ☐ <https://image.dfrobot.com/image/data/KIT0003/DHT11%20datasheet.pdf> -> Datasheet DHT11
- ☐ <https://www.adafruit.com/product/3660> -> Sensor BME680 (temperatura-humidade+gás)

Remover

Figura 8 - Configuração da atividade Consulta de Documentação - exemplo

Após preencher os campos o formador tem disponível um botão *Save* na Inven!RA que permite gravar a configuração, num campo de *properties* definido no registo inicial da atividade. A partir deste momento os campos de registo ficam guardados para posterior *deploy* da atividade para os formandos.

Deploy das atividades: *deploy* da atividade consulta de documentação:

Após configurar todas as atividades, o formador, quando entender, pede à Inven!RA para fazer o *deploy* do IAP. A plataforma devolve ao formador os URL para todas as atividades criadas e configuradas no IAP para que este gira o *deploy* de cada atividade, no LMS, para o formando. Perspetiva-se que, numa fase posterior de desenvolvimento da Inven!RA, este controlo possa ser configurado na plataforma de maneira que seja esta a gerir o momento em que os formandos têm acesso às atividades bem como a verificação do cumprimento de pré-requisitos para a progressão do formando no IAP. No estado atual este controlo é feito pelo formador.

O diagrama de sequência para o *deploy* da atividade de Consulta de Documentação é o que se observa na Figura 9:

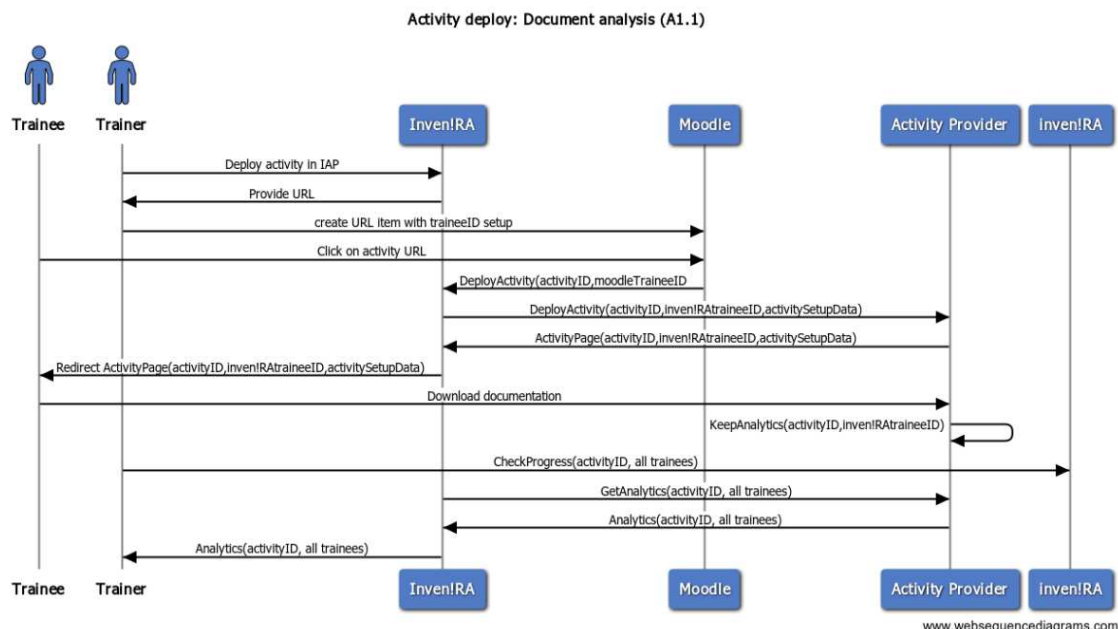


Figura 9 - Diagrama de sequência - deploy atividade Consulta de documentação

Após receber, no LMS, o URL para a atividade, o formando acede para iniciar essa atividade. Neste momento o LMS informa a Inven!RA que o formando com um determinado ID pretende iniciar a atividade com o ID de atividade respetivo. A Inven!RA cria um ID próprio associado ao ID do LMS por forma a garantir a proteção dos dados do formando perante o *Activity Provider*, e envia o pedido à Duarte Lda. que responde com um URL. A Inven!RA redireciona para o formando a página da Duarte Lda. preenchida com os campos de acordo com a configuração enviada no pedido.


Note-se que será através do ID do formando na Inven!RA que a Duarte Lda. poderá manter o registo das interações dos formandos com a página por forma a fornecer ao formador as analíticas desta interação quando este o solicitar à Inven!RA.

Voltando ao exemplo da atividade Consulta de Documentação, a página que o aluno recebe é a que se observa na Figura 10 (códigos nos Anexo 2, 3 e 4). Os campos são preenchidos no momento em que o formando acede à página, com as informações guardadas pela Inven!RA na altura da configuração da atividade. A partir daqui o formando executa a atividade que neste caso consiste em descarregar o descritivo técnico para análise do que vai ter de executar no projeto bem como visualizar ou descarregar os materiais de apoio que o formador sugere para a realização do trabalho.

No momento em que a página é disponibilizada ao aluno é criado um novo objeto na base de dados da Duarte Lda. com os campos para registo das analíticas de progresso na atividade. No Capítulo 3 deste relatório far-se-á a descrição do processo de gestão das analíticas.

inventiveTr@ining

CONSULTA DE DOCUMENTAÇÃO



1. Resumo do descritivo técnico:

Nesta atividade pretende-se planificar, desenhar, configurar e programar uma aplicação com micro-controladores para controlo de parâmetros ambientais numa estufa. Nesta página poderá descarregar o descritivo técnico do projeto bem como alguns documentos de apoio e orientação. Os componentes indicados são meramente sugestões podendo ser utilizados outros desde que forneçam os dados no formato pretendido.

2. Obter o descritivo técnico

Obter

3. Hiperligações para materiais de apoio

☐ Página de referência do Arduino

☐ Datasheet LM35

☐ Datasheet DHT11

☐ Sensor BME680 (temperatura+humidade+gás)

Obter

Figura 10 - Deploy da atividade Consulta de Documentação

Configurar atividade: atividade de configuração de hardware

Esta atividade será lançada na altura em que o aluno irá proceder aos testes de comissionamento e operabilidade do protótipo que desenvolveu, quer em termos de validação do *hardware* (ligações) quer em termos de *firmware* (código). De salientar que o objetivo é testar o código de *firmware* que o aluno programou para cumprir os objetivos pedagógicos do projeto da UFCD.

Assim sendo ao configurar esta atividade o formador disponibiliza um esqueleto de código com as configurações de ligação por Ethernet ou Wi-Fi, dependendo do equipamento entregue ao formando, cabendo a este codificar a recolha dos dados dos sensores, analisar esses dados e atualizar os *outputs* do sistema, de acordo com o descritivo técnico e as instruções fornecidas.

Para além deste código-base o formador deverá fazer *upload* de instruções para o formando, bem como documentação técnica e bibliotecas de sensores, URL para consulta, etc.

O formulário de configuração é o que se pode analisar na Figura 11 da página seguinte (códigos nos Anexo 2, 3 e 4).

Tal como no caso da atividade Consulta de Documentação, o formulário é apresentado ao formador, pela Inven!RA, numa *iframe* HTML.



CRIAR ATIVIDADE - TESTES DE FIRMWARE



1. Resumo:

2. URL para instruções:

3. URL para o download do código-base:

4. URL das hiperligações para bibliotecas

URL + Descritivo do URL:

Lista de hiperligações:

Figura 11 - Página da configuração da atividade de testes de firmware

Na Figura 12 observamos o resultado de uma simulação de configuração de uma atividade de configuração de *hardware* – testes de *firmware*:



CRIAR ATIVIDADE - TESTES DE FIRMWARE



1. Resumo:

Para a realização de testes do trabalho desenvolvido até agora, deverá utilizar o código-base que poderá ser descarregado utilizando o URL abaixo. Para que o protótipo possa ser ligado e os dados enviados deverá proceder de acordo com as instruções que poderá descarregar no link indicado.

2. URL para instruções:

https://www.dropbox.com/s/y43fcj4y3nvhb7/arduino_base_code.ino?dl=0

3. URL para download de código

https://www.dropbox.com/s/by3uzpmyta3x6p2/instr_A7.1.doc?dl=0

4. URL das hiperligações para bibliotecas/documentos técnicos

URL + Descritivo do URL:

Lista de hiperligações:

- ☐ <https://arduinojson.org/v6/doc/installation/> -> Instruções para instalação da biblioteca ArduinoJson
- ☐ https://github.com/adafruit/Adafruit_BMP280_Library -> Instruções para download e utilização da biblioteca do sensor BMP280
- ☐ <https://github.com/adafruit/DHT-sensor-library> -> Instruções para download e utilização da biblioteca do sensor DHT11
- ☐ https://github.com/adafruit/Adafruit_Sensor -> Instruções para download e utilização da biblioteca Adafruit Sensor

Figura 12 - Página de configuração da atividade preenchida

Tal como se preconiza para qualquer atividade configurada na Inven!RA, após preencher os campos do formulário o formador tem disponível na Inven!RA um botão *Save* que permite gravar a configuração. A partir deste momento os campos de registo ficam guardados para posterior *deploy* da atividade para os formandos.

Deploy das atividades: *deploy* da atividade de testes de firmware:

O momento do *deploy* da atividade de testes de *firmware* processa-se da seguinte forma. Na posse do URL fornecido pela plataforma, o formador disponibiliza-o no LMS para os formandos. Estes acedem através do LMS que solicita à Inven!RA a página de *deploy* do *Activity Provider*. Esta página possui o *layout* que pode ser observado na Figura 13 (códigos nos Anexo 2, 3 e 4).

Como já foi referido, esta atividade inclui especificidades inerente à natureza eminentemente prática de implementação. Obviamente que o formando já teve de integrar o hardware, testar os sensores, desenvolver código, etc., conforme descrito no grafo de subactividades. Nesta fase é chegada a altura de proceder aos testes finais. Para isso o formando irá ligar o protótipo que desenvolveu à Internet e enviar dados para que o formador possa ir acompanhando e orientando os trabalhos. Estes dados ficarão armazenados numa base de dados gerida pelo *Activity Provider* que os fornecerá sempre que o formador os solicitar à Inven!RA.



inventiveTr@ining
CONFIGURAÇÃO DE HARDWARE - TESTES DE FIRMWARE

1. Resumo:

Para a realização de testes do trabalho desenvolvido até agora, deverá utilizar o código-base que poderá ser descarregado utilizando o URL abaixo. Para que o protótipo possa ser ligado e os dados enviados deverá proceder de acordo com as instruções que poderá descarregar no link indicado.

2. Instruções para a realização da atividade

Obter

3. Descarregar código-base:

Obter

4. Hiperligações para bibliotecas e documentos de apoio:

- ☐ Instruções para instalação da biblioteca ArduinoJson
- ☐ Instruções para download e utilização da biblioteca do sensor BMP280
- ☐ Instruções para download e utilização da biblioteca do sensor DHT11
- ☐ Instruções para download e utilização da biblioteca Adafruit Sensor

Obter

Figura 13 - Deploy da atividade Testes de firmware

A Figura 14 ilustra o diagrama de sequência da atividade de testes de *firmware*. Para melhor ilustrar a componente que envolve o processo de ligação do formando à base de dados da Duarte Lda, existe um subdiagrama específico para este processo (*Process A*) – Figura 15

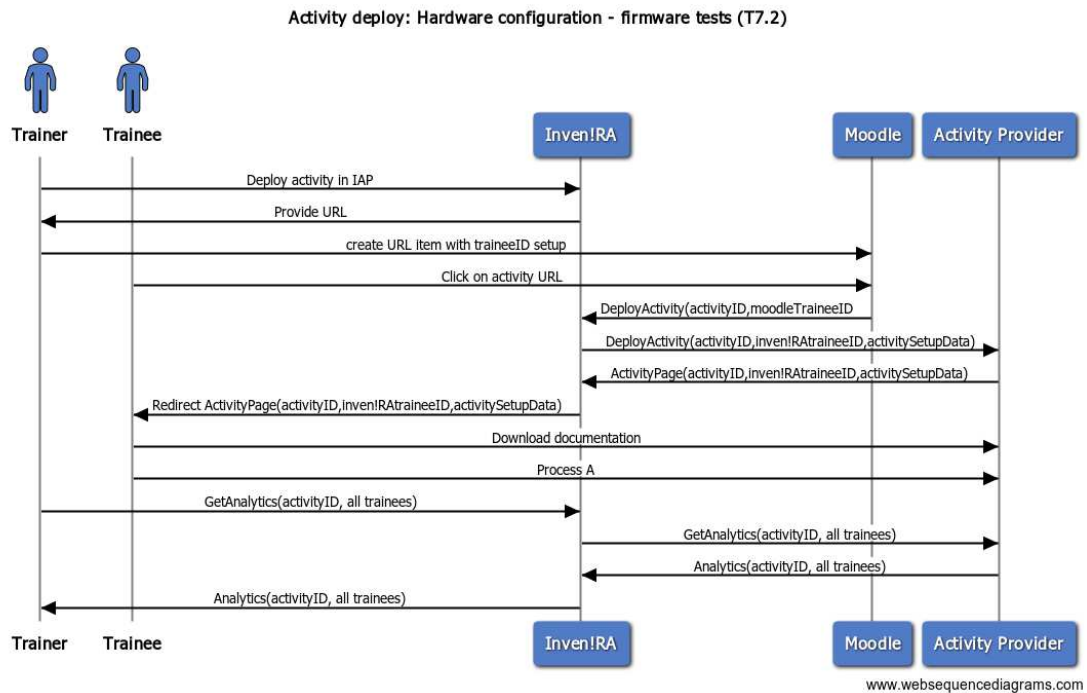


Figura 14 - Diagrama de sequência da atividade Testes de firmware

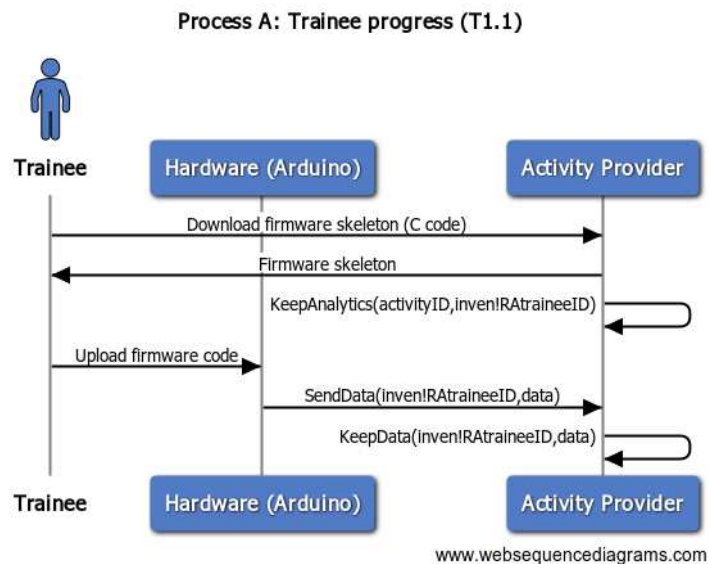


Figura 15 - Diagrama de sequência Process A

Na figura 16 observa-se o *print* de um pacote JSON (exemplo) enviado pelo microcontrolador:


```

from database {
  downloadDocsLibs: [
    {
      desc: 'Instruções para instalação da biblioteca ArduinoJson',
      dload: true
    }
  ],
  studentData: [
    {
      temperatura: [Array],
      'pressão': [Array],
      humidade: [Array],
      alarme: [Array],
      setpoint: [Array],
      inveniraStdID: 1000
    },
    {
      temperatura: [Array],
      'pressão': [Array],
      humidade: [Array],
      alarme: [Array],
      setpoint: [Array],
      inveniraStdID: 1000
    }
  ],
  _id: 5edff325672e995388b4e1de,
  activityID: 1235,
  inveniraStdID: 1000,
  access: true,
  downloadInstr: true,
  downloadCode: false,
  upload: true,
  date: 2020-06-09T20:37:57.763Z,
  __v: 0
}

```

Figura 16 - Pacote JSON recebido no servidor (studentData)

No exemplo da Figura 16 utilizaram-se três sensores de teste que enviam os dados de temperatura do ar, da pressão atmosférica, da humidade relativa e do estado de um alarme que depende de um *setpoint* definido pelo aluno na fase de codificação, conforme as instruções. Todavia estes dados dependerão do contexto do projeto definido no descritivo técnico e será o formando a determinar quais serão.

Este objeto JSON é gerido pela API Node JS que o armazena na MongoDB Atlas associado ao registo com o mesmo ID do formando – Figura 17. O objetivo é controlar o envio de dados para o servidor. Assim, cada pacote está limitado a um número finito de valores de medições dos sensores, neste caso utilizou-se 10 leituras com um intervalo de 2 segundos. Este valor é predefinido no código base de forma a gerar dados suficientes para o formador monitorizar se os objetivos estão a ser atingidos pelo formando e assim efetuar correções. Terá de ter-se em conta, também, a limitada capacidade de memória do microcontrolador de teste.

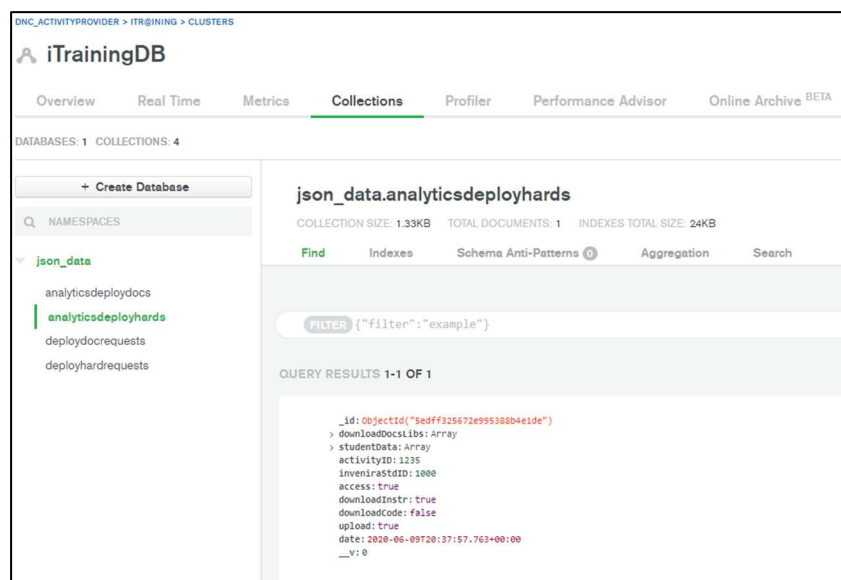


Figura 17 - MongoDB Atlas

O protótipo de testes utilizado possui a seguinte configuração:

- Placa de desenvolvimento Arduino Uno R3 equipada com microcontrolador ATmega328: 8 bits; 32KB de memória FLASH; 2KB SRAM; 1KB EEPROM
- *Shield* Ethernet para Arduino;
- Sensor DTH22;
- Sensor BMP280;

Para gerar o formato JSON recorreu-se à biblioteca *ArduinoJson6*, disponibilizada ao formando para download na fase de *deploy* da atividade.

3. IMPLEMENTAÇÃO DE OUTPUTS - ANALÍTICAS

Na perspetiva do desenvolvimento da *inventiveTr@ining*, a fase que se segue é o desenho e desenvolvimento das ferramentas de *output*. Recorde-se qual o foco principal deste projeto: desenhar e desenvolver componentes para permitir que uma plataforma online em desenvolvimento no INESC TEC possa ser utilizada para acompanhamento de projetos curriculares no âmbito das UFCD de cursos profissionais (nível4) e CET (nível 5), inseridas na área de formação 523-Eletrónica e Automação, de acordo com o Catálogo Nacional de Qualificações.

Quaisquer das atividades e subactividades que resultaram do processo de análise do caso de estudo escolhido, originam um conjunto de dados que permitem ao formador acompanhar o progresso dos formandos e dar resposta a essa evolução intervindo em conformidade e quando necessário (Anexo 1).

Os dados necessários para este processo de acompanhamento e orientação irão resultar do processamento de analíticas que o *Activity Provider* se comprometeu a fornecer à Inven!RA, no momento do registo da atividade, quando esta o solicitar. Tomando como exemplo as duas atividades desenvolvidas neste projeto podemos considerar as algumas analíticas-tipo a fornecer ao formador:

Analíticas quantitativas (*true/false*):

- Alunos que acederam à página de *deploy* da atividade;
- Alunos que descarregaram descritivos técnicos;
- Alunos que descarregaram o código-base;
- Alunos que descarregaram instruções;
- Alunos que descarregaram as bibliotecas necessárias ou documentos técnicos de apoio;
- Alunos que enviaram dados de teste;
- Outras conforme a atividade.

Analíticas quantitativas (*percentagem*):

- Percentagem de documentos descarregados;
- Percentagem de progresso na atividade;
- Outras conforme a atividade.

Analíticas qualitativas:

- Se o aluno descarregou documentos, quais os documentos que descarregou;
- Se o *hardware* do aluno já enviou dados, que dados foram enviados;
- Outras conforme a atividade.

Note-se que as analíticas para casa atividade não são personalizáveis, isto é, o formador não pode criar analíticas, estas são definidas pelo *Activity Provider*, no momento do desenho e implementação da atividade, cabendo ao formador atribuir-lhes o grau de relevância que entender no processo de acompanhamento da aprendizagem.

Sempre que um formando acede ao URL de *deploy* de uma atividade e lhe é disponibilizada a respetiva página, a base de dados da Duarte Lda. regista um objeto com parâmetros *false/true*, e campos do tipo *array* para armazenamento de dados. Observe-se o exemplo da atividade de Consulta de Documentação. Quando a página é visualizada pelo aluno, o objeto criado na base de dados é o seguinte (log do servidor) – Figura 18:

```
{
  docsDesc: [
    { desc: 'Página de referência do Arduino', dLoad: false },
    { desc: 'Datasheet LM35', dLoad: false },
    { desc: 'Datasheet DHT11', dLoad: false },
    { desc: 'Sensor BME680 (temperatura+humidade+gás)', dLoad: false }
  ],
  _id: 5ede83a069a7744374adb8d1,
  activityID: 1234,
  inveniraStdID: 1005,
  access: true,
  downloadDT: false,
  date: 2020-06-08T18:29:52.694Z,
  __v: 0
}
```

Figura 18 - Objeto criado no servidor (Consulta de documentação)

No objeto constam os seguintes campos:

- ID da atividade (*activityID*) = 1234;
- ID do formando na Inven!RA (*inveniraStdID*) = 1005;
- Registo do acesso à página (*access*) = *true* (o formando já acedeu á atividade);
- Registo do *download* do Descritivo Técnico (*downloadDT*) = *false* (o formando ainda não descarregou o documento);
- Registo dos documentos de apoio indicados pelo formador (*docsDesc*) contendo um objeto com o descritivo do recurso e um campo para registo da descarga, no caso *dLoad* = *false* (o formando ainda não descarregou qualquer recurso);

No caso da outra atividade desenvolvida no âmbito deste projeto (Configuração de Hardware), a tipologia é semelhante. Quando a página é visualizada pelo aluno, o objeto criado na base de dados é o seguinte (log do servidor) – Figura 19:

```

from database []
{
  downloadDocsLibs: [
    {
      desc: 'Instruções para instalação da biblioteca ArduinoJson',
      dLoad: false
    },
    {
      desc: 'Instruções para download e utilização da biblioteca do sensor BMP280',
      dLoad: false
    },
    {
      desc: 'Instruções para download e utilização da biblioteca do sensor DHT11',
      dLoad: false
    },
    {
      desc: 'Instruções para download e utilização da biblioteca Adafruit Sensor',
      dLoad: false
    }
  ],
  studentData: [],
  _id: 5ee0dbfcde1bc826d895f544,
  activityID: 1235,
  inveniraStdID: 1008,
  access: true,
  downloadInstr: false,
  downloadCode: false,
  upload: false,
  date: 2020-06-10T13:11:24.523Z,
  __v: 0
}
Objeto criado com sucesso...

```

Figura 19 - Objeto criado no servidor (Configuração de hardware)

No objeto constam os seguintes campos:

- ID da atividade (*activityID*) = 1235;
- ID do formando na Inven!RA (*inveniraStdID*) = 1008;
- Registo do acesso à página (*access*) = *true* (o formando já acedeu à atividade);
- Registo do *download* das instruções (*downloadInstr*) = *false* (o formando ainda não descarregou as instruções);
- Registo do *download* do código base (*downloadCode*) = *false* (o formando ainda não descarregou o esqueleto do código para o projeto);
- Registo se foi efetuado algum *upload* de dados (*upload*) = *false* (o *hardware* do formando ainda não enviou dados);
- Registo dos documentos e bibliotecas de código indicadas pelo formador (*downloadDocsLibs*) contendo um objeto com o descritivo do recurso (*desc*) e um campo para registo da descarga, no caso *dLoad* = *false* (o formando ainda não descarregou qualquer recurso);
- Um campo *studentData* (estrutura do tipo *array*) onde serão guardados os pacotes de dados enviados pelo formando, inicialmente vazio.

A partir do momento em que estes objetos forem criados (note-se que isto acontece no primeiro acesso de um formando a uma atividade), sempre que ocorrer uma interação do formando com a página de *deploy*, independentemente do número de vezes em que a página é acedida, é enviado um pedido para a base de dados atualizar o campo respeitante à interação do formando. Por exemplo, se o formando descarrega o descritivo técnico na atividade Consulta de Documentação, o campo *downloadDT* é atualizado para *true*.

Como já foi referido, a atividade de Configuração de *Hardware* possui a particularidade de permitir que o formando ligue o seu protótipo à Internet e envie pacotes de dados para o servidor da Duarte Lda., o *Activity Provider* neste contexto. Para que isso aconteça e atendendo aos objetivos da UFCD deste caso de estudo, não faz parte dos conteúdos que o formando seja capaz de criar um *Web Server* com o dispositivo que lhe é fornecido. Como tal é disponibilizado pelo formador um código-base ao qual o formando acede via

download na página de *deploy*. A tarefa do formador será programar uma função/método que lê os dados dos sensores e os armazena num *array* (um para cada variável analisada).

O mesmo código-base está preparado para efetuar leituras com intervalos de 1 segundo, num total de 10 em cada corrida do algoritmo, gerar um pacote JSON e enviar num *HTTP request* para o servidor. Esta limitação prende-se, como já referido, com as limitações de memória do microcontrolador normalmente utilizado em contexto de formação profissional (Arduino ou outro compatível) bem como com critérios de razoabilidade no envio de dados para o formador: entende-se que basta uma sequência de 10 leituras para o formador concluir sobre a funcionalidade do código do formador. Todavia este valor é sempre definido pelo formador no código-base.

Processamento de analíticas

Quando o formador entender, acede à Inven!RA e pede o estado do IAP. A plataforma envia um pedido ao *Activity Provider* indicando o ID da atividade ou atividades pretendidas. O servidor responde com os dados recolhidos a partir da sua base de dados (JSON com os parâmetros das analíticas registados com a atividade).

Observemos o exemplo da atividade Consulta de Documentação. Na Figura 20 observa-se o JSON de resposta para a atividade 1234. Note-se que para esta atividade houve dois formandos que acederam à atividade, até ao momento do pedido do formador:

```
[
  {
    "inveniraStdID": 1002,
    "quantAnalytics": [
      {
        "name": "Acesso à atividade",
        "type": "boolean",
        "value": true
      },
      {
        "name": "Download Descritivo Técnico",
        "type": "boolean",
        "value": true
      },
      {
        "name": "Download Documentos Técnicos (%)",
        "type": "percentage",
        "value": "50.00"
      },
      {
        "name": "Progresso na atividade (%)",
        "type": "percentage",
        "value": "66.67"
      }
    ]
  },
  {
    "qualAnalyticsURL": "https://inventivetraining.herokuapp.com/qualAnalyticsDoc/?activityID=1234&inveniraStdID=1002"
  },
  {
    "inveniraStdID": 1003,
    "quantAnalytics": [
      {
        "name": "Acesso à atividade",
        "type": "boolean",
        "value": true
      },
      {
        "name": "Download Descritivo Técnico",
        "type": "boolean",
        "value": true
      },
      {
        "name": "Download Documentos Técnicos (%)",
        "type": "percentage",
        "value": "75.00"
      },
      {
        "name": "Progresso na atividade (%)",
        "type": "percentage",
        "value": "83.33"
      }
    ]
  },
  {
    "qualAnalyticsURL": "https://inventivetraining.herokuapp.com/qualAnalyticsDoc/?activityID=1234&inveniraStdID=1003"
  }
]
```

Figura 20 - JSON de resposta ao pedido de analíticas (Consulta de documentação)

Nesta resposta podemos observar o tipo de analíticas definidas para a atividade. Para além do ID do formando, temos dois campos distintos: *quantAnalytics* e *qualAnalyticsURL*. O primeiro é um *array* de objetos com nome, tipo e valor que constituem um grupo de analíticas quantitativas, e o segundo campo é um URL para uma página da Duarte Lda. que mostra outro tipo de analíticas de carácter qualitativo. Os valores referentes aos campos “Download Documentos Técnicos (%)” e “Progresso na Atividade (%)” são calculados pelo *handler* do servidor que recebe o pedido de analíticas e as devolve à Inven!RA, depois de recolher o estado de cada registo na base de dados.

Caberá à Inven!RA reunir todas as analíticas do IAP numa grelha e mostrar ao formador. Caso o formador pretenda analisar as analíticas qualitativas, no âmbito do processo de acompanhamento do desenvolvimento do IAP, acede ao URL que lhe é fornecido nessa grelha, como se demonstrará no Capítulo 4.

No caso da atividade de Configuração de Hardware, o processo de pedido/resposta de analíticas é semelhante apenas diferindo nos objetos do campo *quantAnalytics* – Figura 21:

```
[
  {
    "inveniraStdID": 1000,
    "quantAnalytics": [
      {
        "name": "Acesso à atividade",
        "type": "boolean",
        "value": true
      },
      {
        "name": "Download Instruções",
        "type": "boolean",
        "value": false
      },
      {
        "name": "Download Código-Base",
        "type": "boolean",
        "value": false
      },
      {
        "name": "Upload dados",
        "type": "boolean",
        "value": false
      },
      {
        "name": "Download Documentos Técnicos (%)",
        "type": "percentage",
        "value": "0.00"
      },
      {
        "name": "Progresso na atividade (%)",
        "type": "percentage",
        "value": "12.50"
      }
    ],
    "qualAnalyticsURL": "https://inventivetraining.herokuapp.com/qualAnalyticsHardConfig/?activityID=1235&inveniraStdID=1000"
  }
]
```

Figura 21 - JSON de resposta ao pedido de analíticas (Configuração de hardware)


4. TESTES DE INTEGRAÇÃO NA PLATAFORMA INVEN!RA

Como já referido neste relatório, o desenvolvimento da plataforma Inven!RA decorreu em paralelo com o projeto *inventiveTr@ining*. Não tendo sido sempre possível compatibilizar na totalidade calendarizações e gerir os trabalhos de forma a permitir o avanço conjunto dos dois projetos, foi decidido, nesta parte final do desenvolvimento, eliminar algumas dependências por forma a agilizar o desenvolvimento individual dos dois trabalhos.

Para efeitos de simulação de operabilidade deste projeto, os pedidos HTTP da Inven!RA foram simulados através da API *Insomnia*, para validar os fluxos de pedidos e respostas bem como para *debug* dos códigos desenvolvidos neste projeto. Estes testes

foram realizados para pedidos de configuração de atividades e *deploy* de atividades por forma a validar os JSON de resposta e a sua conformidade com o formato de dados pré-definido.

Em particular, para simular o *deploy* de analíticas foi criada pelo autor deste projeto uma página de teste que processa o JSON de resposta ao pedido de analíticas da Inven!RA. Na Figura 22 encontra-se o *layout* da página de teste de simulação de analíticas da Inven!RA, para a atividade de Configuração de Hardware – testes de *firmware*:



Aluno	Analítica	Status	Outras analíticas
1000	Acesso à atividade Download Instruções Download Código-Base Upload dados Download Documentos Técnicos (%) Progresso na atividade (%)	SIM NÃO NÃO NÃO 0.00 12.50	http://localhost:5000/qualAnalyticsHardConfig/?activityID=1235&inveniraStdID=1000
1001	Acesso à atividade Download Instruções Download Código-Base Upload dados Download Documentos Técnicos (%) Progresso na atividade (%)	SIM SIM SIM SIM 100.00 100.00	http://localhost:5000/qualAnalyticsHardConfig/?activityID=1235&inveniraStdID=1001
1002	Acesso à atividade Download Instruções Download Código-Base Upload dados Download Documentos Técnicos (%) Progresso na atividade (%)	SIM NÃO NÃO NÃO 0.00 12.50	http://localhost:5000/qualAnalyticsHardConfig/?activityID=1235&inveniraStdID=1002

Figura 22 - Simulação de analíticas (Configuração de hardware)

Neste exemplo verifica-se que três formandos já deram início à atividade. O formador pode observar, para cada formando, o estado da atividade, por exemplo, pode verificar que o formando 1001 já fez *upload* de dados bem como verificar o estado da descarga de recursos. Caso o formador pretenda analíticas mais detalhadas, acede, através do URL do formando respetivo, à página disponibilizada pela Duarte Lda. Observe-se o exemplo do formando 1001. A página evidencia a seguinte informação – Figura 23:






ANÁLÍTICAS DA ATIVIDADE: Configuração de hardware

Listagem de documentos descarregados:

- Instruções do projeto

Listagem de código descarregado:

- Código-base
- Instruções para instalação da biblioteca ArduinoJson
- Instruções para download e utilização da biblioteca do sensor BMP280
- Instruções para download e utilização da biblioteca do sensor DHT11
- Instruções para download e utilização da biblioteca Adafruit Sensor

Dados enviados pelo formando:

Figura 23 - Página de analíticas qualitativas (Configuração de Hardware)

Neste caso o formando já descarregou todos os documentos técnicos, já descarregou o código-base e as instruções. Essa informação detalhada é evidenciada nesta página. Se o formador pretender observar os dados enviados pelo projeto do formando, pode escolher entre visualizar todos os pacotes recebidos no servidor ou optar pelo último pacote recebido. Na Figura 24 poder ver-se o resultado da opção “Obter pacote mais recente”. Caso opte por visualizar todos os dados a página mostra uma tabela para cada pacote guardado no servidor – Figura 25

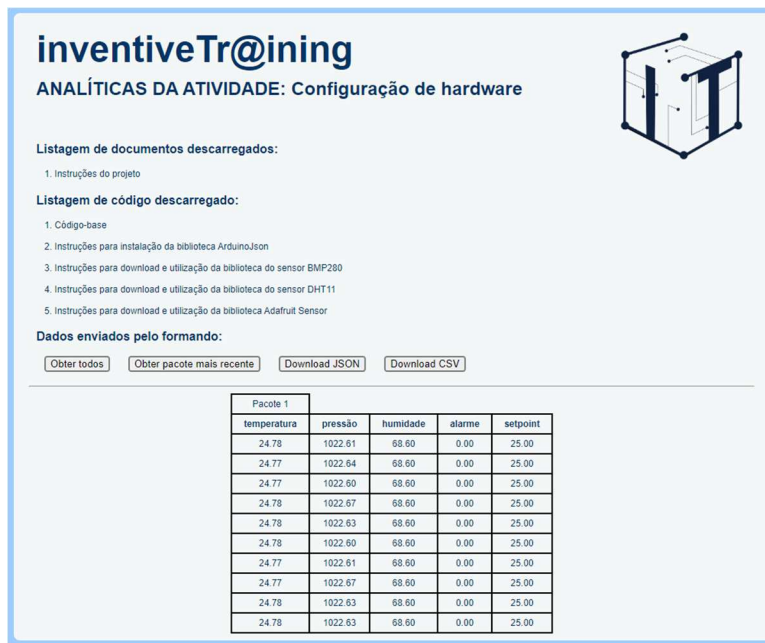


Figura 24 - Página de análíticas qualitativas –ver pacote de dados mais recente

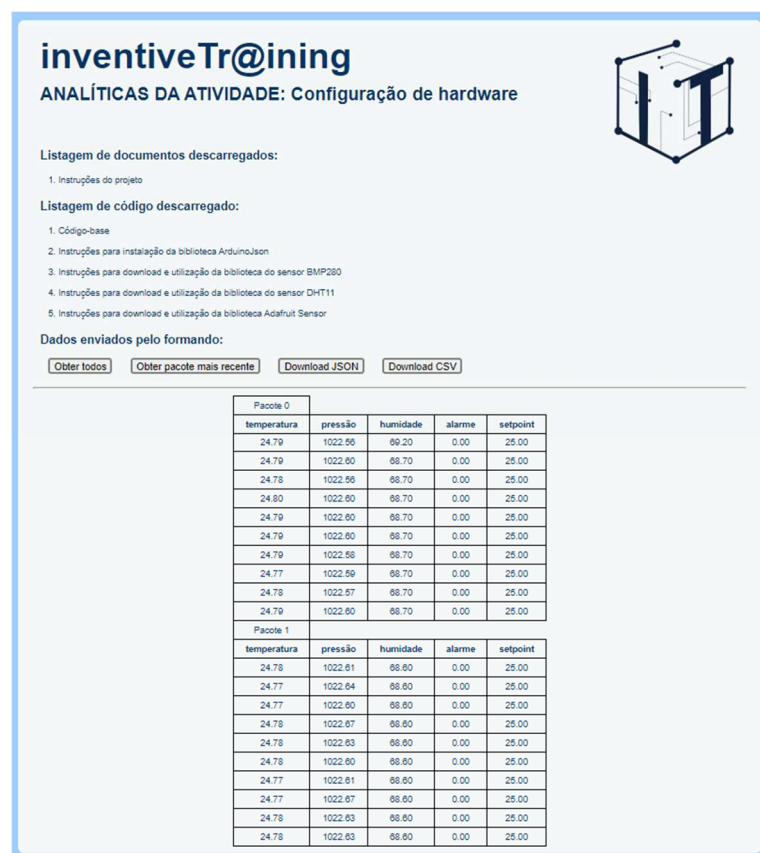


Figura 25 - Página de análíticas qualitativas –ver todos

Na eventualidade do formador pretender guardar os dados enviados pelo formando para posterior análise/processamento, existem dois botões que pode utilizar para fazer *download* de um ficheiro nos formatos JSON ou CSV.

No caso da atividade Consulta de Documentação, a simulação de analíticas terá o seguinte *layout* – Figura 26:



The screenshot shows the Inven!RA interface. At the top, there's a header with the Inven!RA logo. Below it, a dropdown menu is set to 'Consulta de documentação'. Another dropdown shows the value '1234'. The main heading is 'ATIVIDADE: Consulta de documentação'. Below this is a table with analytics data.

Aluno	Analítica	Status	Outras analíticas
1002	Acesso à atividade Download Descritivo Técnico Download Documentos Técnicos (%) Progresso na atividade (%)	SIM SIM 50.00 66.67	https://inventivetraining.herokuapp.com/qualAnalyticsDoc/?activityID=1234&inveniraStdID=1002
1003	Acesso à atividade Download Descritivo Técnico Download Documentos Técnicos (%) Progresso na atividade (%)	SIM SIM 75.00 83.33	https://inventivetraining.herokuapp.com/qualAnalyticsDoc/?activityID=1234&inveniraStdID=1003
1000	Acesso à atividade Download Descritivo Técnico Download Documentos Técnicos (%) Progresso na atividade (%)	NÃO SIM 0.00 16.67	https://inventivetraining.herokuapp.com/qualAnalyticsDoc/?activityID=1234&inveniraStdID=1000
1001	Acesso à atividade Download Descritivo Técnico Download Documentos Técnicos (%) Progresso na atividade (%)	SIM NÃO 0.00 16.67	https://inventivetraining.herokuapp.com/qualAnalyticsDoc/?activityID=1234&inveniraStdID=1001
1004	Acesso à atividade Download Descritivo Técnico Download Documentos Técnicos (%) Progresso na atividade (%)	SIM SIM 50.00 66.67	https://inventivetraining.herokuapp.com/qualAnalyticsDoc/?activityID=1234&inveniraStdID=1004

Figura 26 - Simulação de analíticas (Consulta de Documentação)

Cada URL abre uma página preenchida com os documentos descarregados pelo respetivo formando. Na Figura 27 observa-se a página de analíticas qualitativas para um determinado formando.

Apesar desta atividade ser relativamente simples, considera-se que, mesmo assim, fica demonstrada a flexibilidade quer da plataforma Inven!RA quer do serviço inventiveTr@ining na gestão das analíticas, o que contribui consideravelmente para a prova de conceito de ambas as ferramentas.



The screenshot shows the inventiveTr@ining interface. The header includes the logo and the title 'ANÁLÍTICAS DA ATIVIDADE: Consulta de documentação'. On the right, there is a 3D cube icon. The main content area is titled 'Listagem de documentos descarregados pelo formando:' and contains a list of three items.

inventiveTr@ining	
ANÁLÍTICAS DA ATIVIDADE: Consulta de documentação	
Listagem de documentos descarregados pelo formando:	
1.	Descritivo Técnico
2.	Página de referência do Arduino
3.	Datasheet LM35

Figura 27 - Página de analíticas qualitativas (Consulta de Documentação)

Este processo de simulação foi fundamental para o teste e *debug* do código e decorreu sempre em articulação com a equipa de desenvolvimento da Inven!RA. Posteriormente foi possível testar a integração dos serviços da Duarte Lda. na Inven!RA. Em relação ao processo de configuração das atividades, na *Figura 28* podemos visualizar a página de configuração da atividade de Consulta de Documentação integrada na Inven!RA (*iframe*):

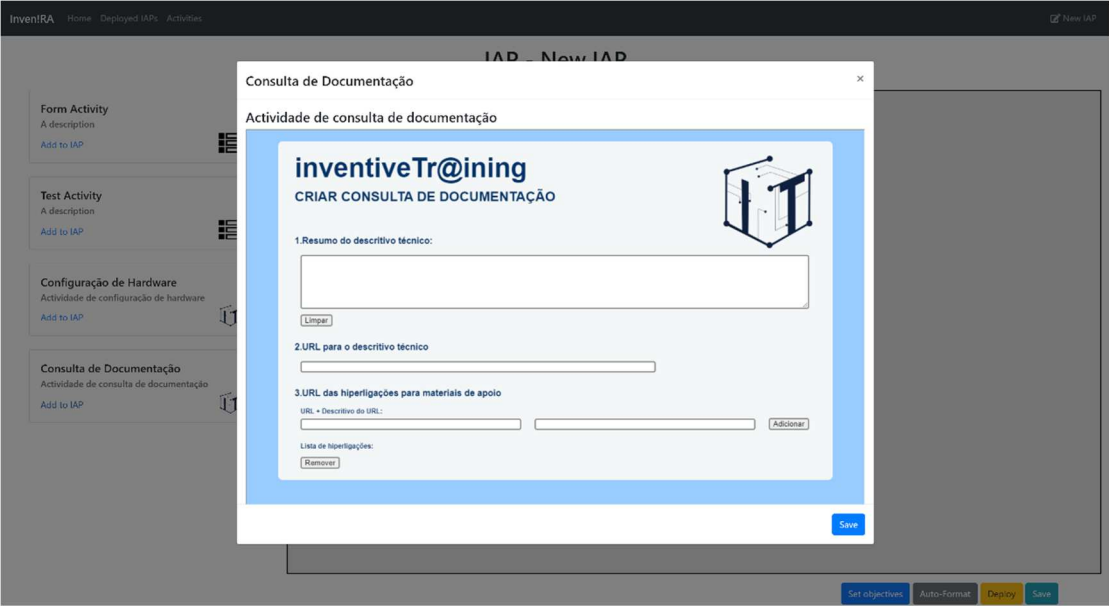


Figura 28 - UI de configuração de atividade. Créditos: Tiago Cruzeiro – Inven!RA

Depois de configuradas as atividades no IAP, o formador recebe, da Inven!RA, indicação dos URL (*Deployment URL*) para cada uma delas, e poderá lançar as atividades para os formandos no LMS – *Figura 29*:

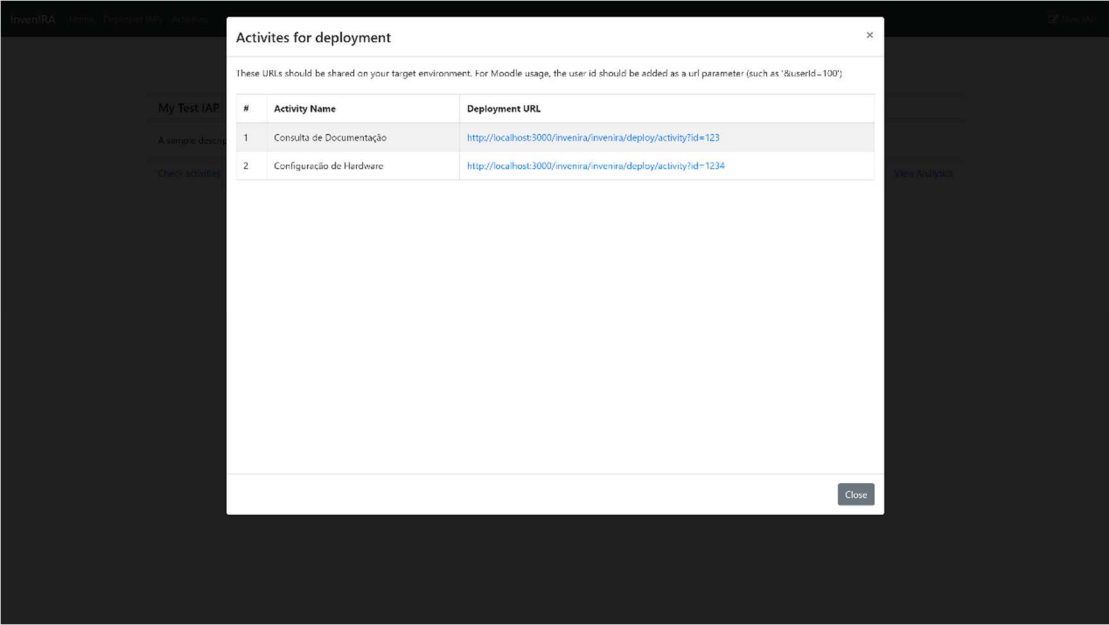


Figura 29 - UI de deploy de atividades. Créditos: Tiago Cruzeiro – Inven!RA

No seguimento do pedido do estado do IAP, a Inven!RA pede as analíticas ao *Activity Provider* que responde enviando um ficheiro JSON com a estrutura já descrita (*Figuras 19 e 20*). Após processamento o formador pode visualizar as analíticas de cada atividade na seguinte UI– *Figura 30*:

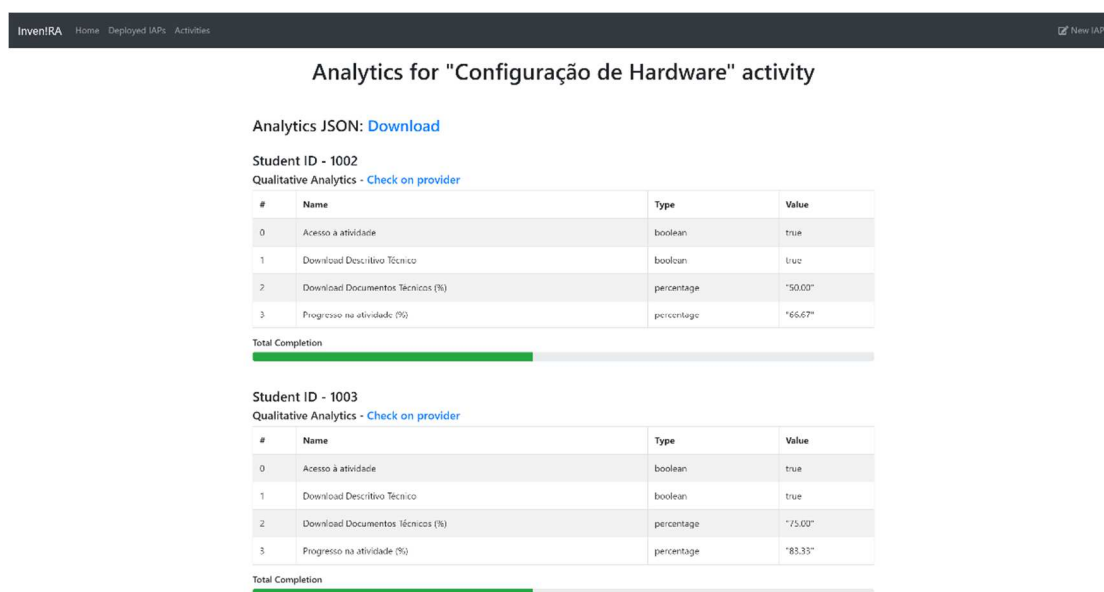


Figura 30 - UI consulta de analíticas. Créditos: Tiago Cruzeiro – Inven!RA

Caso pretenda aceder às analíticas qualitativas poderá utilizar a hiperligação “*Check on Provider*” que lhe mostrará a respetiva página do *Activoty Provider* para cada formando (*Figuras 23 e 27*)

Em relação aos testes de integração das atividades de *deploy*, uma vez que os mesmos não implicam qualquer processamento de *frontend* da Inven!RA, não são aqui evidenciadas figuras. Neste âmbito a intervenção da Inven!RA limita-se a redirecionar para o formando o URL fornecido pela Duarte Lda. no momento do pedido de *deploy*, para, que este possa ter acesso aos dados configurados e realizar as atividades.

Não foi realizado um teste com integração do LMS que permitisse testar o processo de resposta ao pedido do formando para aceder à atividade. Todavia o serviço da parte da Duarte Lda. foi validado no respetivo teste já descrito acima. Assim, todos os testes de que envolvem a integração dos dois projeto, especificamente no fornecimento dos serviços preconizados na definição de objetivos, foram concluídos com sucesso validando a operação dos serviços da inventiveTr@ining no contexto de utilização da Inven!RA.

Prevê-se que este projeto tenha continuidade, após esta fase, estando prevista o prosseguimento do desenvolvimento das restantes atividades em articulação com a nova equipa internacional que irá continuar o projeto de investigação Inven!RA.

5. TESTES COM UTILIZADORES

A etapa de verificação e validação por testes do produto final, implementado no âmbito deste projeto, sucede à etapa de codificação e tem como objetivo fundamental garantir

que todos os pressupostos estão cumpridos em termos de desenvolvimento. Esta é uma missão de todos os intervenientes envolvidos, através do cumprimento de um plano de ações e verificações (testes) que validem se o produto está conforme o desenho inicial, e até para efeito de introdução de melhorias de usabilidade.

Uma primeira etapa – verificação – foi realizada pelo autor e desenvolvedor do projeto e garante que a aplicação está alinhada com as especificações iniciais. A esta etapa seguiu-se a fase de validação, um ponto crítico que em engenharia de software significa a execução de um plano de testes pelos clientes/utilizadores finais. Para esta fase pretendia-se, inicialmente, conseguir testar toda a integração com utilizadores, com o objetivo de afinar os processos de interligação dos diferentes componentes, e trabalhar as analíticas através da simulação da interação dos utilizadores com os componentes do sistema para melhor se definir a ligação dos respetivos ID nas plataformas Moodle e Inven!RA. Não será de mais salientar que por razões de proteção de dados dos estudantes, o *Activity Provider* não sabe quem é o formando. A Inven!RA recebe esses dados, nos pedidos de interação no LMS, e associa um ID próprio (*inveniraStdID*). É este ID que é enviado nos pedidos ao *provider*, em conjunto com o ID das atividades (*activityID*). Com base nesta informação o serviço *inventiveTr@ining* processa os pedidos, mantém os registos na sua base de dados e processa as respostas necessárias.

Um modelo de testes adequado incluiria um grupo de formandos e formadores, um plano com a definição de atividades distintas e um conjunto de interações que pudessem evidenciar erros ou falhas do projeto. Perante a dificuldade em organizar um plano de testes intensivos da *inventiveTr@ining*, em boa parte devido à pandemia do novo Coronavírus (suspensão de atividades letivas presenciais), optou-se pela realização de testes locais onde o autor do projeto assume o papel de todos os intervenientes, nomeadamente:

- *Activity Designer*: quem desenha as atividades, da parte da Duarte Lda.
- *Learning Designer* ou concetor de planos de atividades inventivas (IAP): formador que neste papel configura as atividades para os IAP;
- *Formador*: na sua função nativa interage com a Inven!RA e com o LMS, disponibiliza as atividades para os formandos, acompanha e monitoriza o progresso dos estudantes e age em conformidade (orienta, alerta, avalia...)
- *Formando*: estudante que executa as atividades no IAP e interage com o LMS (Moodle) e com a Duarte Lda. (envio de dados).

Na tentativa de obter a granularidade máxima possível, realizou-se o seguinte plano de testes:

Testes unitários:

Foram realizados essencialmente testes de “Caixa Aberta” ao longo do processo de desenvolvimento, com o objetivo de testar os componentes isoladamente, nomeadamente a sua resposta à entrada de dados. Foram executados com recurso à ferramenta de teste e *debug* de API *Insomnia*, que permite a execução de pedidos HTTP com corpo de dados JSON, conforme especificação da Inven!RA, em integração com utilização de três *browsers* diferentes (*Edge*, *Firefox* e *Chrome*), Visual Studio Code 2017 e linha de comandos do servidor (visualização dos *log* do servidor após processamento das ações). Estes testes também incluíram testes de entrada de dados diretamente nos formulários de configuração (*browser*) para verificação da conformidade dos campos de

registo dos dados e geração de ficheiro JSON. Pese embora o desenho dos serviços preconize que o processamento destes dados seja realizado pela Inven!RA, é necessário que os campos fiquem com os dados disponíveis conforme especificado no registo da atividade. Para os testes que envolvem a utilização de *hardware* (atividade de Configuração de hardware – testes de *firmware*) foi utilizado um protótipo semelhante ao que é utilizado em contexto de formação presencial em laboratório, descrito na secção 2.6. Os resultados obtidos ao longo do desenvolvimento foram sempre alinhados com os objetivos de integração na Inven!RA e foram determinantes para efetuar correções e afinações na dinâmica de fluxo de dados.

Teste de integração:

Os testes de integração verificam se os componentes funcionam conjuntamente. Optou-se pela alternativa *Top-down* que pressupõe uma certa ordem na sequência de testes, em função da abrangência dos componentes. Neste caso e como só foram desenvolvidas duas atividades do grafo de subactividades inicial, optou-se por testar a sequência de atividades pela ordem de desenvolvimento tal como serão utilizadas num contexto real (só poderá existir *deploy* das atividades depois da sua configuração). Foram obtidos resultados compatíveis com o esperado, nomeadamente ao nível do fluxo de informação entre componentes (ID de atividades e de estudantes) que são determinantes para a correta identificação dos diferentes intervenientes no sistema, manutenção da base de dados e de analíticas, etc.)

Testes de aceitação:

Os testes de aceitação validam se os requisitos dos utilizadores finais estão implementados no produto final. Trata-se de um conjunto de validações suportados por testes dinâmicos, neste caso com a Inven!RA atuando como cliente que disponibiliza os serviços ao utilizador final (formador/formando). Embora todo o desenvolvimento tenha sido realizado em ambiente local, nesta fase migrou-se para um alojamento na plataforma em *cloud* para permitir os testes de integração com a Inven!RA. Foi executado um conjunto de testes completo, em articulação com a quipá Inven!RA, que cobriu todos os serviços disponibilizadas pelo projeto *inventiveTr@ining*, nas vertentes já identificadas. Esta série de testes permitiu evidenciar as potencialidades desta ferramenta e listar as falhas/melhorias que serão identificadas mais à frente neste documento. Face às considerações descritas e aos fatores evidenciados consideram-se atingidos os objetivos da fase de testes.

6. CRONOGRAMA

Definido na proposta inicial do projeto *inventiveTr@ining*, o cronograma de desenvolvimento das atividades constituiu uma peça determinante na obtenção do cumprimento dos objetivos pré-definidos.

Salienta-se que, atendendo à natureza da UC de Projeto de Engenharia, que decorre em simultâneo com outras UC da licenciatura em Engenharia Informática, um rigoroso cumprimento da calendarização, evitando ou compensando desvios imprevistos, permitiu que o projeto decorresse sem incidentes significativos e conduziu à conclusão do desenvolvimento dentro dos prazos estipulados.

Não será de mais salientar a estreita articulação da equipa de desenvolvimento da Inve!RA, que facilitou em permanência os avanços necessários à conclusão deste trabalho com sucesso.

O cronograma do projeto – Figura 31 – encontra-se indicado com informação colorida das atividades de projeto total ou parcialmente desenvolvidas.

	Mês	Março			Abril					Maio				Junho		
		Semana														
		2	3	4	1	2	3	4	5	1	2	3	4	1	2	3
ATIVIDADES	ESPECIFICAÇÃO	ESP														
	Descrever a atividade pedagógica a planificar.															
	Decompô-la num grafo de subactividades (plano de atividades).															
	Identificar que dados devem essas atividades fornecer ao professor															
	Identificar de que forma se podem obter esses dados.															
	IMPLEMENTAÇÃO DO INPUT				INP											
	Implementar formulários Web para especificação de cada atividade.															
	Implementar outros componentes de recolha de dados.															
	INTEGRAÇÃO					INT										
	Integrar na plataforma Inven!RA.															
	IMPLEMENTAÇÃO DO OUTPUT									OUT						
	Implementar visualizações de monitorização e acompanhamento.															
	TESTES												TST			
	Aplicar e testar com utilizadores.															
	DOCUMENTAÇÃO														DOC	
	Relatório final.															

	Concretizado
	Parcialmente concretizado
	Por concretizar

Figura 31 - Cronograma do projeto inventiveTr@ining

7. CONCLUSÕES

Considera-se que o desenvolvimento do projeto inventiveTr@ining decorreu conforme o cronograma e planificação inicial.

Para o sucesso do desenvolvimento do projeto, foi determinante assumir a inexequibilidade da implementação de todas as atividades que resultaram do trabalho realizado e descrito nas secções 2.1 a 2.3, e a escolha de apenas duas atividades-tipo para desenvolver em maior profundidade. Considera-se que, a concretização destas duas

atividades, provam o conceito pretendido. Efetivamente fica demonstrado que a *inventiveTr@ining* é uma ferramenta de usabilidade simples e cumpre perfeitamente os dois objetivos subjacentes à proposta inicial:

- Criar uma ferramenta de apoio à formação técnica de alunos do ensino profissional, permitindo-lhes desenvolver projetos práticos em contexto “fora da escola” como estratégia de mitigação da reduzida carga horária letiva nas componentes técnicas;
- Integração do projeto na plataforma *Inven!RA* que constitui uma excelente ferramenta de integração de atividades e gestão de planos de aula/atividades inventivas (IAP) na medida em que permite integrar atividades de qualquer *Activity Provider* num plano único, acompanhar a execução do plano e atuar em conformidade com os objetivos do IAP.

Para o desenvolvimento e implementação deste projeto e para materializar os serviços disponibilizados pelo *Activity Provider* (a Duarte Lda, assim batizada na primeira reunião de orientação), foi necessário adquirir e consolidar conhecimentos complementares aos adquiridos nas Unidades Curriculares da licenciatura, nomeadamente instalação e configuração de um servidor HTTP em Node JS e aprofundamento de conhecimentos e técnicas de programação em JavaScript, HTML e CSS.

Um sólido entendimento do conceito de *Web service* e dos conceitos de *RESTful API* e pedidos *HTTP* facilitou a integração do fluxo e gestão de pedidos/respostas entre os componentes do sistema, criando processos simples e eficientes de comunicação entre os diferentes módulos.

As maiores dificuldades encontradas situaram-se ao nível da articulação entre o projeto *inventiveTr@ining* e o projeto *Inven!RA*, na medida da necessidade de compatibilização dos dois desenvolvimentos para efeitos de integração.

Durante o desenvolvimento deste trabalho de projeto, nomeadamente na fase final de testes foram identificadas algumas melhorias que podem ser introduzidas numa fase posterior, tendo a ver com detalhes importantes, mas não críticos nem impactantes na funcionalidade dos componentes já desenvolvidos. De seguida apresenta-se uma lista de possíveis melhorias identificadas, no âmbito das vertentes já desenvolvidas:

- Atividade de Consulta de Documentação
 - Para esta atividade e dada a sua natureza não foram identificadas melhorias substanciais. Todavia, uma das melhorias a introduzir poderá ser a Duarte Lda. disponibilizar um serviço de alojamento para onde os formadores enviam toda a documentação, nomeadamente a de produção própria como descritivos técnicos e outros documentos relevantes, em vez de recorrer a serviços externos de *cloud hosting* como Dropbox, Google Drive ou equivalentes.
- Atividade de Configuração de Hardware – Testes de *firmware*
 - Para esta atividade seria interessante implementar uma forma de evitar que um formando possa enviar dados com o ID de outro formando. No sistema atual o formando deverá incluir manualmente no seu código o ID que lhe é fornecido no momento em que faz o *download* do código-

- base. Isto pode originar situações de fraude que podem ser evitadas se for desenhada uma forma de o sistema saber que o pacote de dados recebido no *Activity Provider* é proveniente de um formando e do equipamento que lhe foi consignado, eventualmente usando o *MAC Address* do equipamento. Em alternativa ser possível gerar o ID do formando automaticamente, adicionando-o ao código-base no momento do *download*. Para isso seria necessário que o código-base fosse editado pelo formador na página de configuração ou então carregado para o servidor do *provider* em vez de somente disponibilizado o URL para posterior *download*.
- Seria interessante incluir no *payload* do pedido enviado pelo projeto do formando, não só os pacotes de dados, mas também o próprio código desenvolvido pelo formando evitando o envio posterior através de outra forma. Uma vez que a compilação do código não é feita na plataforma de desenvolvimento (Arduino) não é fácil aceder ao ficheiro específico para o enviar. As limitações de memória do Arduino também são um fator restritivo para esta operação. O uso de plataformas do tipo Raspberry Pi, seriam, para este fim, muito mais apelativas.

Em termos de trabalhos futuros, considera-se que numa primeira fase impõe-se desenvolver as atividades em falta para completar a sequência de tarefas que constituem os objetivos pedagógicos da UFCD escolhida para este caso de estudo. Uma das valências a implementar relaciona-se com os questionários autorreflexivos e de validação das aprendizagens previstos para a transição entre atividades.

Pensa-se desenvolver uma página que permita ao profissional docente criar questionários do tipo escolha múltipla ou resposta Verdadeiro/Falso. Esses questionários bem como as respetivas soluções e cotações ficariam registados na base de dados da Duarte Lda. para posterior *deploy* aos estudantes. Quando o estudante submeter o questionário haveria verificação automática das respostas na API da Duarte Lda. para retorno da classificação obtida. Independentemente destes mecanismos já estarem disponíveis em diversas plataformas de LMS, eliminar esta dependência poderia ser uma mais valia no contexto de atividades feitas à medida inerentes ao projeto de investigação Inven!RA.

Outros trabalhos podem ser equacionados tais como as páginas da *inventiveTr@inig* possuírem dispositivos próprios para, por exemplo, desenho de fluxogramas ou de esquemas elétricos, quer de implementação própria ou através de inserção de *plugins* de ferramentas recorrentes já existentes.

Todavia é fundamental manter o carácter distintivo da plataforma integradora Inven!RA, que não pretendendo constituir-se como concorrente de plataformas de LMS existentes, ambiciona oferecer serviços Web para criação de planos letivos inventivos atrativos e estimuladores quer para os docentes quer para os estudantes, facilitar a sua gestão e o processo de ensino e aprendizagem a formadores e formandos.

8. BIBLIOGRAFIA E RECURSOS WEB

Bibliografia:

Dukket, J. (2011) *HTML & CSS Design and build websites* (1st edition). Indianapolis, Indiana: John Wiley & Sons, Inc.

Brown, E. (2016) *Learning JavaScript: JavaScript essentials for modern application development* (3rd edition). Sabastopol, CA: O'REILLY Media, Inc.

Pereira, A., Poupa C. (2004) *Linguagens Web* (5ª edição). Lisboa: Edições Sílabo

Brown, E., *Web Development with Node and Express Leveraging the JavaScript Stack* (2nd edition). Sabastopol, CA: O'REILLY Media, Inc.

Massé, M., *REST API Design Rulebook* (1st edition). Sabastopol, CA: O'REILLY Media, Inc.

Guerreiro, S. (2015) *Introdução à Engenharia de Software* (1ª edição). Lisboa: FCA

Outros recursos Web:

JSON schema validator: <https://jsonlint.com/>

NODE JS API Reference Documentation: <https://nodejs.org/en/docs/>

npm (Node Package Manager) Documentation: <https://docs.npmjs.com/>

W3Schools: <https://www.w3schools.com/>

9. ACEITAÇÃO DO ORIENTADOR

No Anexo 7 encontra-se o documento de aceitação deste relatório, pelo orientador do projeto inventiveTr@ining, Doutor Leonel Morgado.

ANEXO 1 : MAPA DE ATIVIDADES E SUBATIVIDADES

ANEXO 1 - Mapa de atividades e subatividades										
No.	Timeline	Atividade	Subatividade	Descrição da atividade	Notas	Espaço	Deadline	Aula	Dados relevantes para a atuação do professor	Atuação do professor
	Dia 1				Apresentado o processo da atividade, documentação disponibilizada online	Sala de aula	50 min.	x		
1	Dias 1-7	A1	A1.1	ANÁLISE DE DOCUMENTAÇÃO Analisar o descritivo técnico da atividade. QUIZ_1: Questionário sobre a análise do descritivo técnico da atividade.	Questionário autorreflexivo, não determina avanço na atividade. Serve para o aluno refletir sobre se está preparado para iniciar a atividade pedagógica.	Fora da sala de aula	Dia 7		Dia 3: Lista de alunos que não abriram o descritivo técnico Dia 3: Lista de alunos que abriram o descritivo técnico, mas não submeteram o QUIZ_1.	Encorajar os alunos a abrir o descritivo técnico. Relembrar a existência do QUIZ_1.
2		A2	A2.1	PREENCHIMENTO DE QUADRO QUIZ_2: Questionário sobre a estrutura e detalhe de um mapa de quantidades.	Questionário autorreflexivo e sumativo. Serve para o aluno avaliar se compreende a importância da elaboração de um mapa completo de quantidades e para o professor avaliar se o aluno está preparado para criar e preencher o mapa.				Dia 5: Lista de alunos que não preencheu o QUIZ_2	Relembrar que o QUIZ_2 é sumativo.
3			A2.2	PREENCHIMENTO DE QUADRO Elaborar mapa de quantidades dos recursos identificados como necessários. Submeter o mapa de quantidades apurado.					Dia 7: Lista de alunos com aprovação no QUIZ_2 Dia 7: Mapas de quantidades entregues	Análise dos mapas de quantidades submetidos. Emitir recomendações.
	Dia 8				Realização de ponto de situação das atividades + Consignação de recursos técnicos	Sala de aula	50 min.	x		
4	Dias 8-12	A3	A3.1	DESENHO DE HARDWARE QUIZ_3: Questionário sobre diagramas de ligações em eletrônica.	Questionário autorreflexivo e sumativo. Serve para o aluno autoavaliar as suas competências em matéria de elaboração de diagramas de ligações e para o professor diagnosticar se o aluno está preparado para os criar.	Fora da sala de aula	Dia 12		Dia10: Lista de alunos que não preencheu o QUIZ_3	Relembrar que o QUIZ_3 é sumativo.
5			A3.2	DESENHO DE HARDWARE Desenhar o mapa de ligações (<i>hardware</i>) em <i>software</i> específico. Submeter o diagrama.					Dia 11: Lista de alunos com aprovação no QUIZ_3. Dia11: Lista de alunos que submeteu o diagrama.	Análise dos diagramas submetidos. Emitir recomendações.
	Dia 13				Realização de ponto de situação das atividades	Sala de aula	50 min.	x		
6	Dia 13-16	A4	A4.1	DESENHO DE FLUXOGRAMA QUIZ_4: Questionário sobre estrutura e concepção de fluxogramas.	Questionário autorreflexivo e sumativo. Serve para o aluno autoavaliar se é capaz de traduzir num diagrama de fluxo, as especificidades do <i>firmware</i> que tem de criar para o microcontrolador, e para o professor diagnosticar se o aluno compreendeu a sequência de passos que são necessários para implementar o <i>firmware</i> (sequência, decisões, obtenção de dados, etc.)	Fora da sala de aula	Dia 16		Dia15: Lista de alunos que não preencheu o QUIZ_4	Relembrar que o QUIZ_4 é sumativo.
7			A4.2	DESENHO DE FLUXOGRAMA Desenhar um fluxograma do algoritmo a implementar. Submeter o fluxograma.					Dia 16: Lista de alunos com aprovação no QUIZ_4. Dia16: Lista de alunos que submeteu o diagrama.	Análise dos fluxogramas submetidos. Emitir recomendações.
	Dia 17				Realização de ponto de situação das atividades	Sala de aula	50 min.	x		
8	Dia 17-26	A5	A5.1	INTEGRAÇÃO DE HARDWARE Integrar o hardware de acordo com diagrama desenvolvido em T3		Fora da sala de aula	Dia 26			
9		A6	A6.1	PROGRAMAÇÃO DE COMPONENTES DE SOFTWARE QUIZ_5: Questionário sobre bases de desenvolvimento de software.	Questionário autorreflexivo e sumativo. Serve para o aluno autoavaliar se compreende as bases de programação e algoritmia para programar o seu microcontrolador. Para o professor, serve para diagnosticar o nível de preparação do aluno para a concretização da subarefa.				Dia 20: Lista de alunos que não preencheu o QUIZ_5.	Relembrar que o QUIZ_5 é sumativo.
10			A6.2	PROGRAMAÇÃO DE COMPONENTES DE SOFTWARE Desenvolver o código para o microcontrolador (baseado em C++) Submeter o código.					Dia 22: Lista de alunos com aprovação no QUIZ_5. Dia22: Lista de alunos que submeteu o código.	Avaliar a qualidade e funcionalidade do código Emitir recomendações. Fornecer o esqueleto do código para envio de dados via web.
11		A7	A7.1	CONFIGURAÇÃO DE HARDWARE QUIZ_6: Questionário sobre tipos de testes de funcionamento aplicáveis ao contexto e sobre o tipo de dados que deverão estar a ser gerados. Definir lista de testes a realizar.	Questionário autorreflexivo. Serve para o aluno refletir sobre se tem noção da importância dos testes de funcionamento e sobre que dados são gerados pelo sistema e o que fazer com essa informação. O professor avalia a capacidade do aluno em identificar os testes necessários para a validação do sistema.				Dia 23: Lista de alunos que não preencheu o QUIZ_6.	Relembrar a existência do QUIZ_6.
12			A7.2	CONFIGURAÇÃO DE HARDWARE Fazer o upload do <i>firmware</i> ; iniciar testes de comissionamento e operabilidade.					Dia 25: Alunos que estão com o projeto online a emitir dados.	Vizualiza online os dados gerados pelo protótipo do aluno. Emitir recomendações. Sugere melhorias
13			A7.3	CONFIGURAÇÃO DE HARDWARE Refletir sobre o comportamento do protótipo (<i>hardware</i> e <i>software</i>) e melhorias.					Dia 27: Alunos que estão online e implementaram melhorias.	Vizualiza online os dados gerados pelo protótipo do aluno.
14		A8	A8.1	ELABORAÇÃO DE RELATÓRIO Iniciar uma fase reflexão/autoavaliação.					Dia 21: Lista de alunos que iniciaram a sua reflexão/autoavaliação (preparação para o relatório final).	Lembrar que devem iniciar as suas reflexões para auxiliar no relatório final.
	Dia 27				Realização de ponto de situação das atividades	Sala de aula	50 min.	x		
15	Dia 27-31	A9	A9.1	ELABORAÇÃO DE RELATÓRIO QUIZ_7: Questionário sobre estrutura e conteúdo do relatório.	Questionário autorreflexivo e sumativo. Serve para o aluno avaliar se compreende a estrutura de um relatório técnico e qual o conteúdo adequado. O professor faz o diagnóstico da preparação do aluno para a realização de um relatório aplicado ao contexto.	Fora da sala de aula	Dia 31		Dia 28: Lista de alunos que não preencheu o QUIZ_7.	Relembrar que o QUIZ_7 é sumativo.
16			A9.2	ELABORAÇÃO DE RELATÓRIO Elaborar o relatório final. Submeter o relatório final.					Dia 31: Lista de alunos que submeteu o relatório final.	Vizualiza os relatórios dos alunos Emitir sugestões e sugere melhorias
	Dia 32				Realização de balanço da atividade	Sala de aula	50 min.	x		

ANEXO 2 : CÓDIGOS HTML FRONTEND

```

1  <!--
2  Projet: Projeto LEI
3  Author: Duarte Cota
4  Date: 2020
5  Script: HTML deploy analytics activity Get documentation
6  -->
7
8  <!DOCTYPE html>
9
10 <html>
11   <head>
12     <meta charset="UTF-8">
13     <title>iTr@ining</title>
14     <link rel="icon" type="image/png" href="../img/it_logo_white.png">
15     <link href="../css/style.css" type="text/css" rel="stylesheet"/>
16     <script type="text/javascript" src="../js/script_analyticsDoc.js"></script>
17   </head>
18   <!--ao iniciar executar a função de script() principal -->
19   <body onload = "script();">
20     <div class = "container">
21       <div id = "header">
22         <h1>inventiveTr@ining
23           <a href="/"></a>
24         </h1>
25       </div>
26       <h2>ANALÍTICAS DA ATIVIDADE: Consulta de documentação</h2>
27       <br><br>
28       <h3>Listagem de documentos descarregados pelo formando:</h3>
29       <div id = list > </div>
30
31     </body>
32 </html>

```

```

1  <!--
2  Projet: Projeto LEI
3  Author: Duarte Cota
4  Date: 2020
5  Script: HTML deploy analytics activity Hardware configuration - test firmware
6  -->
7
8  <!DOCTYPE html>
9
10 <html>
11   <head>
12     <meta charset='UTF-8'>
13     <title>iTr@ining</title>
14     <link rel='icon' type='image/png' href='../img/it_logo_white.png'>
15     <link href='../css/style.css' type='text/css' rel='stylesheet' />
16     <script type='text/javascript'
17       src='../js/script_analyticsHardConfig.js'></script>
18   </head>
19   <!--ao iniciar executar a função de script() principal -->
20   <body onload = 'script();'>
21     <div class = 'container'>
22       <div id = 'header'>
23         <h1>inventiveTr@ining
24         <a href='/'><img src='../img/it_logo.png'></a>
25       </h1>
26     </div>
27     <h2>ANALÍTICAS DA ATIVIDADE: Configuração de hardware</h2>
28     <br><br>
29     <h3>Listagem de documentos descarregados:</h3>
30     <div id = listDoc > </div>
31     <h3>Listagem de código descarregado:</h3>
32     <div id = listCode > </div>
33     <h3>Dados enviados pelo formando:</h3>
34     <div>
35       <button id=all> Obter todos </button>
36       <button id=last> Obter pacote mais recente </button>
37       <button id=fileDownloadJSON> Download JSON </button>
38       <button id=fileDownloadCSV> Download CSV </button>
39     </div>
40     <hr>
41     <div id='tableData'></div>
42   </body>
43 </html>

```

```

1  <!--
2  Projet: Projeto LEI
3  Author: Duarte Cota
4  Date: 2020
5  Script: HTML config activity Consult Documentation
6  -->
7
8  <!DOCTYPE html>
9
10 <html>
11   <head>
12     <meta charset="UTF-8">
13     <title>iTr@ining</title>
14     <link rel="icon" type="image/png" href="../img/it_logo_white.png">
15     <link href="../css/style.css" type="text/css" rel="stylesheet"/>
16     <script type="text/javascript" src="../js/script_1.js"></script>
17   </head>
18   <!--ao iniciar executar a função de script principal -->
19   <body onload = "script();" >
20     <div class = "container">
21       <div id="header">
22         <h1>inventiveTr@ining
23           <a href="/"></a>
24         </h1>
25       </div>
26       <h2>CRIAR CONSULTA DE DOCUMENTAÇÃO</h2>
27       <br><br>
28       <h3>1.Resumo do descritivo técnico:</h3>
29       <div>
30         <textarea type="textarea" id="desc" onfocus=this.value = '' rows="4"
31           cols="115"></textarea>
32       </div>
33       <div>
34         <button id=clear_button onclick=
35           "document.getElementById('desc').value = ''> Limpar </button>
36       </div>
37       <br>
38       <h3>2.URL para o descritivo técnico</h3>
39       <div>
40         <input type="text" id="desc_tec" name="desc_tec" size="100">
41       </div>
42       <br>
43       <h3>3.URL das hiperligações para materiais de apoio</h3>
44       <div >
45         <p id= "title_links"><b>URL + Descritivo do URL:</b></p>
46         <input type="text" id="links" name="links" size="60">
47         <input type="text" id="links_desc" name="links" size="60">
48         <button id=add_button> Adicionar </button>
49       </div>
50       <div id = "list_header">
51         <p><b>Lista de hiperligações:</b></p>
52       </div>
53       <div id = "list">
54       </div>
55       <div>
56         <button id=rem_button> Remover </button>
57       </div>
58     </body>
59 </html>

```

```

1  <!--
2  Projet: Projeto LEI
3  Author: Duarte Cota
4  Date: 2020
5  Script: HTML config activity Hardware configuration - test firmware
6  -->
7
8  <!DOCTYPE html>
9
10 <html>
11   <head>
12     <meta charset="UTF-8">
13     <title>iTr@ining</title>
14     <link rel="icon" type="image/png" href="../img/it_logo_white.png">
15     <link href="../css/style.css" type="text/css" rel="stylesheet"/>
16     <script type="text/javascript" src="../js/script_3.js"></script>
17   </head>
18   <!--ao iniciar executar a função de script principal -->
19   <body onload = "script();" >
20     <div class = "container">
21       <div id="header">
22         <h1>inventiveTr@ining
23           <a href="/"></a>
24         </h1>
25       </div>
26       <h2>CRIAR ATIVIDADE - TESTES DE FIRMWARE</h2>
27       <br><br>
28       <h3>1. Resumo:</h3>
29       <div>
30         <textarea type="textarea" id="desc" onfocus=this.value = '' rows="4"
31           cols="115"></textarea>
32       </div>
33       <div>
34         <button id=clear_button onclick=
35           "document.getElementById('desc').value = ''> Limpar </button>
36       </div>
37       <br>
38       <h3>2.URL para instruções:</h3>
39       <div>
40         <input type="text" id="desc_instr" name="desc_instr" size="100">
41       </div>
42       <br>
43       <h3>3.URL para descarga de código</h3>
44       <div >
45         <input type="text" id="desc_code" name="links" size="100">
46       </div>
47       <br>
48       <h3>4.URL das hiperligações para bibliotecas/documentos técnicos</h3>
49       <div >
50         <p id= "title_links"><b>URL + Descritivo do URL:</b></p>
51         <input type="text" id="links" name="links" size="60">
52         <input type="text" id="links_desc" name="links" size="60">
53         <button id=add_button> Adicionar </button>
54       </div>
55       <br>
56       <div id = "list_header">
57         <p><b>Lista de hiperligações:</b></p>
58       </div>
59       <div id = "list">
60       </div>
61       <div>
62         <button id=rem_button> Remover </button>
63       </div>
64     </body>
65 </html>

```

```

1  <!--
2  Projet: Projeto LEI
3  Author: Duarte Cota
4  Date: 2020
5  Script: HTML deploy activity Consult documentation
6  -->
7
8  <!DOCTYPE html>
9
10 <html>
11   <head>
12     <meta charset="UTF-8">
13     <title>iTr@ining</title>
14     <link rel="icon" type="image/png" href="../img/it_logo_white.png">
15     <link href="../css/style.css" type="text/css" rel="stylesheet"/>
16     <script type="text/javascript" src="../js/script_2.js"></script>
17   </head>
18   <!--ao iniciar executar a função de script() principal -->
19   <body onload = "script();">
20     <div class = "container">
21       <div id="header">
22         <h1>inventiveTr@ining
23           <a href="/"></a>
24         </h1>
25       </div>
26       <h2>CONSULTA DE DOCUMENTAÇÃO</h2>
27       <br><br>
28       <h3>1.Resumo do descritivo técnico:</h3>
29       <div>
30         <textarea type="textarea" id="desc" onfocus=this.value='' rows="4"
31           cols="115"></textarea>
32       </div>
33       <br>
34       <h3>2.Obter o descritivo técnico</h3>
35       <div>
36         <button id=getURL_button> Obter </button>
37       </div>
38       <br>
39       <h3>3.Hiperligações para materiais de apoio</h3>
40       <div class="box" id="box2">
41       </div>
42       <br>
43       <button id=getLinks_button> Obter </button>
44     </div>
45   </body>
46 </html>

```

```

1  <!--
2  Projet: Projeto LEI
3  Author: Duarte Cota
4  Date: 2020
5  Script: HTML deploy activity Hardware configuration - test firmware
6  -->
7
8  <!DOCTYPE html>
9
10 <html>
11   <head>
12     <meta charset="UTF-8">
13     <title>iTr@ining</title>
14     <link rel="icon" type="image/png" href="../img/it_logo_white.png">
15     <link href="../css/style.css" type="text/css" rel="stylesheet"/>
16     <script type="text/javascript" src="../js/script_4.js"></script>
17   </head>
18   <!--ao iniciar executar a função de script() principal -->
19   <body onload = "script();">
20     <div class = "container">
21       <div id="header">
22         <h1>inventiveTr@ining
23           <a href="/"></a>
24         </h1>
25       </div>
26       <h2>CONFIGURAÇÃO DE HARDWARE - TESTES DE FIRMWARE</h2>
27       <br><br>
28       <h3>1. Resumo:</h3>
29       <div>
30         <textarea type="textarea" id="desc" onfocus=this.value='' rows="4"
31           cols="115"></textarea>
32       </div>
33       <br>
34       <h3>2. Instruções para a realização da atividade</h3>
35       <div>
36         <button id=getInstr_button> Obter </button>
37       </div>
38       <br>
39       <h3>3. Descarregar código-base:</h3>
40       <div>
41         <button id=getCode_button> Obter </button>
42       </div>
43       <br>
44       <h3>4. Hiperligações para bilbliotecas e documentos de apoio:</h3>
45       <div class = "box" id = "box3">
46         <div>
47           <br>
48           <button id=getLinks_button> Obter </button>
49         </div>
50       </div>
51   </body>
52 </html>

```



```

1  <!--
2  Projet: Projeto LEI
3  Author: Duarte Cota
4  Date: 2020
5  Script: HTML Inven!RA analytics simulation
6  -->
7
8  <!DOCTYPE html>
9
10 <html>
11   <head>
12     <meta charset='UTF-8'>
13     <title>iTr@ining</title>
14     <link rel='icon' type='image/png' href='../img/it_logo_white.png'>
15     <link href='../css/style_invenira.css' type='text/css' rel='stylesheet' />
16     <script type='text/javascript' src='../js/script_invenira.js'></script>
17   </head>
18   <!--ao iniciar executar a função de script() -->
19   <body onload = 'script();'>
20     <div id = 'header'>
21       <h1>Inven!RA</h1>
22     </div>
23     <div>
24       <select id = 'options1'>
25         <option value=''></option>
26         <option value='Consulta de documentação'>Consulta de
27           documentação</option>
28         <option value='Configuração de hardware'>Configuração de
29           hardware</option>
30       </select>
31     </div>
32     <br><br>
33     <div>
34       <select id = 'options2' onchange='getActID(this.id)'>
35       </select>
36     </div>
37     <br><br>
38     <h2 id = 'name'> ATIVIDADE: </h2>
39     <br><br>
40     <table id='table' >
41       <thead>
42         <tr>
43           <th id='aluno'>Aluno</th>
44           <th id='analitica'>Analítica</th>
45           <th id='status'>Status</th>
46           <th id='analQuant'>Outras analíticas</th>
47         </tr>
48       </thead>
49     </table>
50   </body>
51 </html>

```

ANEXO 3 : CÓDIGOS JAVASCRIPT FRONTEND

```

1  /*
2  Projet: Projeto LEI
3  Author: Duarte Cota
4  Date: 2020
5  Script: JavaScript show analytics activity Get documentation
6  */
7
8  //main function
9  function script() {
10     const queryString = window.location.search;
11     const urlParams = new URLSearchParams(queryString);
12     activityID = urlParams.get('activityID');
13     inveniraStdID = urlParams.get('inveniraStdID');
14     getData();
15 }
16
17 function getData() {
18     fetch('https://inventivetraining.herokuapp.com/api/analyticsGetDocRoute/' +
19     activityID + '/' + inveniraStdID)
20         .then(res => res.json())
21         .then(data => processData(data))
22         .catch(function (error) {
23             alert('Request failed', error)
24         });
25 }
26
27 function processData(data) {
28     console.log(data);
29     if(data.length==0)
30         document.getElementById('listDoc').innerHTML += '    Nenhum documento
31         descarregado' + '<br>' + '<br>';
32     else {
33         for (var i=0; i<data.length; i++)
34             document.getElementById('list').innerHTML += (i+1) + '. ' + data[i] +
35             '<br>' + '<br>';
36     }
37 }
38

```

```

1  /*
2  Projet: Projeto LEI
3  Author: Duarte Cota
4  Date: 2020
5  Script: Javascript show analytics activity Hardware configuration - test firmware
6  */
7
8  //main function
9  function script() {
10     const queryString = window.location.search;
11     const urlParams = new URLSearchParams(queryString);
12     activityID = urlParams.get('activityID');
13     inveniraStdID = urlParams.get('inveniraStdID');
14     getData();
15 }
16
17 //get data from database
18 function getData() {
19     fetch('https://inventivetraining.herokuapp.com/api/analyticsGetHardRoute/' +
20 activityID + '/' + inveniraStdID)
21     .then(res => res.json())
22     .then(data => processData(data))
23     .catch(function (error) {
24         alert('Request failed', error)
25     });
26 }
27
28 //process data
29 function processData(data) {
30     console.log(data);
31     if(data.downloadInstr.length == 0){
32         document.getElementById('listDoc').innerHTML += '    Nenhum documento
33 descarregado'+<br>' + '<br>';
34     }
35     else{
36         for (var i=0; i<data.downloadInstr.length; i++){
37             document.getElementById('listDoc').innerHTML += (i+1) + '. ' +
38 data.downloadInstr[i] + '<br>' + '<br>';
39         }
40     }
41     if(data.downloadCode.length == 0){
42         document.getElementById('listCode').innerHTML += '    Nenhum código
43 descarregado'+<br>' + '<br>';
44     }
45     else{
46         for (var i=0; i<data.downloadCode.length; i++){
47             document.getElementById('listCode').innerHTML += (i+1) + '. ' +
48 data.downloadCode[i] + '<br>' + '<br>';
49         }
50     }
51     if(data.studentData.length == 0){
52         const getAll_button = document.getElementById('all')
53         getAll_button.addEventListener('click', function()
54         {
55             document.getElementById('tableData').innerHTML = ' ';
56             document.getElementById('tableData').innerHTML += '    Não há dados para
57 mostrar'+<br>' + '<br>';
58         });
59         const getLast_button = document.getElementById('last')
60         getLast_button.addEventListener('click', function()
61         {
62             document.getElementById('tableData').innerHTML = ' ';
63             document.getElementById('tableData').innerHTML += '    Não há dados para
64 mostrar'+<br>' + '<br>';
65         });
66     }
67     else{
68         const getAll_button = document.getElementById('all')
69         getAll_button.addEventListener('click', function()
70         {
71             showAll(data);
72         });
73         const getLast_button = document.getElementById('last')
74         getLast_button.addEventListener('click', function()
75         {

```

```

67         showLast(data);
68     });
69
70     const getDownloadJSON_button = document.getElementById('fileDownloadJSON')
71     getDownloadJSON_button.addEventListener('click', function()
72     {
73         onDownloadJSON(data.studentData);
74     });
75
76     const getDownloadCSV_button = document.getElementById('fileDownloadCSV')
77     getDownloadCSV_button.addEventListener('click', function()
78     {
79         onDownloadCSV(data.studentData);
80     });
81     }
82 }
83
84 //show all data packs
85 function showAll(data){
86     document.getElementById('tableData').innerHTML = ' ';
87     var table = document.createElement('TABLE');
88     for(var pack in data.studentData){
89         var len = data.studentData[pack].length;
90         var keys = Object.keys(data.studentData[pack]);
91         var aux=[];
92         for(var i in keys){
93             if(keys[i] != 'inveniraStdID')
94                 aux.push(keys[i]);
95         }
96         var columnCount = aux.length
97         var row = table.insertRow(-1);
98         var cell = row.insertCell(-1);
99         cell.style.textAlign = 'center';
100        cell.innerHTML = 'Pacote ' + pack;
101        row.appendChild(cell);
102        var row = table.insertRow(-1);
103        for (var i = 0; i < columnCount; i++) {
104            var headerCell = document.createElement('TH');
105            headerCell.innerHTML = aux[i];
106            row.appendChild(headerCell);
107            var len = data.studentData[pack][aux[0].valueOf()].length;
108        }
109        row = table.insertRow(-1);
110        for (var j = 0; j < len; j++) {
111            for(params in aux) {
112                var cell = row.insertCell(-1);
113                cell.style.textAlign = 'center';
114                cell.innerHTML =
115                    parseFloat(data.studentData[pack][aux[params].valueOf()][j]).toFixed(2)
116                    + '<br>';
117            }
118            row = table.insertRow(-1);
119        }
120        var tableData = document.getElementById('tableData');
121        tableData.innerHTML = ' ';
122        tableData.appendChild(table);
123    }
124 }
125
126 //show just the last pack
127 function showLast(data){
128     var lastPack = data.studentData.length-1;
129     document.getElementById('tableData').innerHTML = ' ';
130     var table = document.createElement('TABLE');
131     var len = data.studentData[lastPack].length;
132     var keys = Object.keys(data.studentData[lastPack]);
133     var aux=[];
134     for(var i in keys){
135         if(keys[i] != 'inveniraStdID')
136             aux.push(keys[i]);
137     }
138     var columnCount = aux.length;
139     var row = table.insertRow(-1);

```

```

138     var cell = row.insertCell(-1);
139     cell.style.textAlign = 'center';
140     cell.innerHTML = 'Pacote ' + lastPack;
141     row.appendChild(cell);
142     var row = table.insertRow(-1);
143     for (var i = 0; i < columnCount; i++) {
144         var headerCell = document.createElement('TH');
145         headerCell.innerHTML = aux[i];
146         row.appendChild(headerCell);
147         var len = data.studentData[lastPack][aux[0].valueOf()].length;
148     }
149     row = table.insertRow(-1);
150     for (var j = 0; j < len; j++) {
151         for (params in aux) {
152             var cell = row.insertCell(-1);
153             cell.style.textAlign = 'center';
154             cell.innerHTML =
                parseFloat(data.studentData[lastPack][aux[params].valueOf()][j]).toFixed(2)
                + '<br>';
155         }
156         row = table.insertRow(-1);
157     }
158
159     var tableData = document.getElementById('tableData');
160     tableData.innerHTML = '';
161     tableData.appendChild(table);
162
163 }
164
165 //on click to download JSON
166 function onDownloadJSON(data) {
167     download(JSON.stringify(data, undefined, 4), 'dadosAlunoJSON.json', 'text/plain');
168 }
169
170 //on click to download CSV
171 function onDownloadCSV(data) {
172     download(toCSV(data), 'dadosAlunoCSV.csv', 'text/csv;charset=utf-8;');
173 }
174
175 //download function
176 function download(content, fileName, contentType) {
177     const a = document.createElement('a');
178     const file = new Blob([content], { type: contentType });
179     a.href = URL.createObjectURL(file);
180     a.download = fileName;
181     a.click();
182 }
183
184 //convert JSON to CSV
185 function toCSV (data){
186     const items = data;
187     const replacer = (key, value) => value === null ? '' : value
188     const header = Object.keys(items[0])
189     let csv = items.map(row => header.map(fieldName =>
190     JSON.stringify(row[fieldName], replacer)).join(','))
191     csv.unshift(header.join(','))
192     csv = csv.join('\r\n')
193     return csv
194 }

```

```

1  /*
2  Projet: Projeto LEI
3  Author: Duarte Cota
4  Date: 2020
5  Script: JavaScript config activity Consult documentation
6  */
7
8  //properties Inven!RA registration
9  var summary = '';
10 var dtURL = '';
11 var docsURLs = [];
12
13 //aux fields
14 var count_links = 0;
15
16 //main script
17 function script()
18 {
19     const description_box = document.getElementById('desc')
20     description_box.addEventListener('change', function()
21     {
22         summary = document.getElementById('desc').value;
23     });
24
25     const dt_doc_box = document.getElementById('desc_tec')
26     dt_doc_box.addEventListener('change', function()
27     {
28         dtURL = document.getElementById('desc_tec').value;
29     });
30
31     const add_button = document.getElementById('add_button')
32     add_button.addEventListener('click', function()
33     {
34         var lnk = document.getElementById('links').value;
35         var desc = document.getElementById('links_desc').value;
36         if(desc != '') {
37             var linkObj = new Object();
38             linkObj.lnk = lnk;
39             linkObj.desc = desc;
40             docsURLs.push(linkObj);
41             addLink(linkObj);
42             document.getElementById('links').value = '';
43             document.getElementById('links_desc').value = '';
44         }
45         else
46             alert('Tem de preencher uma descrição!');
47     });
48     const rem_button = document.getElementById('rem_button')
49     rem_button.addEventListener('click', function()
50     {
51         deleteFromLinks();
52     });
53 }
54
55 /***** aux functions*****/
56
57 //output last URL link from list
58 function addLink(linkObj) {
59     var myDiv = document.getElementById('list');
60     // creating checkbox element
61     var checkbox = document.createElement('input');
62     checkbox.type = 'checkbox';
63     checkbox.name = 'name';
64     checkbox.value = 'value';
65     checkbox.id = count_links;
66     // creating label for checkbox
67     var label = document.createElement('label');
68     // assigning attributes for the created label tag
69     label.htmlFor = 'id';
70     // appending the created text to the created label tag
71     var lb = linkObj.lnk + ' -> ' + linkObj.desc;
72     label.appendChild(document.createTextNode(lb));
73     //appending the checkbox and label to div

```

```

74     myDiv.appendChild(checkbox);
75     myDiv.appendChild(label);
76     document.getElementById('list').innerHTML += '<br>';
77     count_links = count_links + 1;
78     document.getElementById('links').innerHTML = '';
79     document.getElementById('links_desc').innerHTML = '';
80 }
81
82 //deletes last URL from list
83 function deleteFromLinks() {
84     var linksAux = new Array();
85     var count = document.getElementById('list').children;
86     for(var i=0; i<count.length; i++) {
87         if(count[i].type == 'checkbox' && count[i].type == 'checkbox') {
88             if(!count[i].checked) {
89                 linksAux.push(docsURLs[count[i].id]);
90             }
91         }
92     }
93     count_links = 0;
94     document.getElementById('list').innerHTML = '';
95     docsURLs=[];
96     for(var j=0; j<linksAux.length; j++) {
97         docsURLs.push(linksAux[j]);
98         addLink(docsURLs[j]);
99     }
100     alert('Hiperligações eliminadas!');
101 }
102
103 /*
104 //create JSON file from form data - used in tests phase
105 function createJASON(summary,dtURL,docsURLs)
106 {
107     var obj = new Object();
108     obj.summary = summary;
109     obj.dtURL = dtURL;
110     obj.docsURLs = docsURLs;
111     myJSON = JSON.stringify(obj);
112     console.log(myJSON);
113     const options = {
114         method: 'POST',
115         headers: {
116             'Content-type': 'application/json'
117         },
118         mode:'cors',
119         body: myJSON
120     };
121     fetch('http://tiagoc.xyz/invenira/activity',options)
122     .then(function (response) {
123         return response.text();
124     })
125     .then(function (text) {
126         console.log('Request successful:', text);
127     })
128     .catch(function (error) {
129         log('Request failed', error)
130     });
131 }
132
133 //GET JSON objects from server - used if needed
134 function getJSON() {
135     fetch('http://tiagoc.xyz/invenira/activity')
136     .then(function(response) {
137         return response.json()
138     })
139     .then(function(data) {
140         console.log(data);
141     })
142 }*/

```



```

1  /*
2  Projet: Projeto LEI
3  Author: Duarte Cota
4  Date: 2020
5  Script: JavaScript config activity Hardware configuration - test firmware
6  */
7
8  //properties Inven!RA registration
9  var summary = '';
10 var codeURL = '';
11 var instrURL = '';
12 var libsURLs = [];
13
14 //aux fields
15 var count_links = 0;
16
17 //main script
18 function script()
19 {
20     const description_box = document.getElementById('desc')
21     description_box.addEventListener('change', function()
22     {
23         summary = document.getElementById('desc').value;
24         console.log(summary);
25     });
26
27     const code_doc_box = document.getElementById('desc_code')
28     code_doc_box.addEventListener('change', function()
29     {
30         codeURL = document.getElementById('desc_code').value;
31         console.log(codeURL);
32     });
33
34     const instr_doc_box = document.getElementById('desc_instr')
35     instr_doc_box.addEventListener('change', function()
36     {
37         instrURL = document.getElementById('desc_instr').value;
38         console.log(instrURL);
39     });
40
41     const add_button = document.getElementById('add_button')
42     add_button.addEventListener('click', function()
43     {
44         var lnk = document.getElementById('links').value;
45         var desc = document.getElementById('links_desc').value;
46         if(desc != '') {
47             var linkObj = new Object();
48             linkObj.lnk = lnk;
49             linkObj.desc = desc;
50             libsURLs.push(linkObj);
51             addLink(linkObj);
52             document.getElementById('links').value = '';
53             document.getElementById('links_desc').value = '';
54         }
55         else
56             alert('Tem de preencher uma descrição!');
57     });
58
59     const rem_button = document.getElementById('rem_button')
60     rem_button.addEventListener('click', function()
61     {
62         deleteFromLinks();
63     });
64 }
65
66 /***** aux functions*****/
67
68 //outputs last URL link from list
69 function addLink(linkObj) {
70     var myDiv = document.getElementById('list');
71     // creating checkbox element
72     var checkbox = document.createElement('input');
73     checkbox.type = 'checkbox';

```

```

74     checkbox.name = 'name';
75     checkbox.value = 'value';
76     checkbox.id = count_links;
77     // creating label for checkbox
78     var label = document.createElement('label');
79     // assigning attributes for the created label tag
80     label.htmlFor = 'id';
81     // appending the created text to the created label tag
82     var lb = linkObj.lnk + ' -> ' + linkObj.desc;
83     label.appendChild(document.createTextNode(lb));
84     //appending the checkbox and label to div
85     myDiv.appendChild(checkbox);
86     myDiv.appendChild(label);
87     document.getElementById('list').innerHTML += '<br>';
88     count_links = count_links + 1;
89     document.getElementById('links').innerHTML = '';
90     document.getElementById('links_desc').innerHTML = '';
91     createJASON(summary,codeURL,instrURL,libsURLs);
92 }
93
94 //deletes last URL from list
95 function deleteFromLinks() {
96     var linksAux = new Array();
97     var count = document.getElementById('list').children;
98     for(var i=0; i<count.length; i++) {
99         if(count[i].type == 'checkbox' && count[i].type == 'checkbox') {
100             if(!count[i].checked) {
101                 linksAux.push(libsURLs[count[i].id]);
102             }
103         }
104     }
105     count_links = 0;
106     document.getElementById('list').innerHTML = '';
107     libsURLs=[];
108     for(var j=0; j<linksAux.length; j++) {
109         libsURLs.push(linksAux[j]);
110         addLink(libsURLs[j]);
111     }
112     alert('Hiperligações eliminadas!');
113 }

```

```

1  /*
2  Projet: Projeto LEI
3  Author: Duarte Cota
4  Date: 2020
5  Script: JavaScript deploy activity Get documentation
6  */
7
8  //empty fields for JSON input
9  var description = '';
10 var dtDoc = '';
11 var linkToDT = '';
12 var links = [];
13 var docsURLs = [];
14 var activityID
15 var inveniraStdID;
16
17 //aux fields
18 var count_links = 0;
19
20 //main function
21 function script() {
22     //query on URL params
23     const queryString = window.location.search;
24     console.log(queryString);
25     const urlParams = new URLSearchParams(queryString);
26     inveniraStdID = urlParams.get('inveniraStdID');
27     console.log(inveniraStdID);
28     activityID = urlParams.get('activityID');
29     console.log(activityID);
30
31     //call function to get data to deploy
32     getData();
33
34     const getURL_button = document.getElementById('getURL_button')
35     getURL_button.addEventListener('click', function()
36     {
37         openDoc();
38         updateDB(activityID,inveniraStdID,'dtURL');
39     });
40
41     const getLinks_button = document.getElementById('getLinks_button')
42     getLinks_button.addEventListener('click', function()
43     {
44         openLinks();
45     });
46
47 }
48
49 //show URL description
50 function showURLs(link) {
51     links.push(link); //adiciona ao array local para abrir
52     var myDiv = document.getElementById('box2');
53     //creating checkbox element
54     var checkbox = document.createElement('input');
55     checkbox.type = 'checkbox';
56     checkbox.name = 'name';
57     checkbox.value = 'value';
58     checkbox.id = count_links;
59     //creating label for checkbox
60     var label = document.createElement('label');
61     //assigning attributes for the created label tag
62     label.htmlFor = 'id';
63     //appending the created text to the created label tag
64     label.appendChild(document.createTextNode(link.desc));
65     //appending the checkbox and label to div
66     myDiv.appendChild(checkbox);
67     myDiv.appendChild(label);
68     document.getElementById('box2').innerHTML += '<br>';
69     count_links = count_links + 1;
70 }
71
72 //open URL
73 function openLinks() {

```

```

74     var count_children = document.getElementById('box2').children;
75     docsURLs=[];
76     var docArray = function(desc){
77         this.desc = desc;
78     };
79     for(var i=0; i<count_children.length; i++) {
80         if(count_children[i].type == 'checkbox') {
81             if(count_children[i].checked) {
82                 window.open(links[count_children[i].id].lnk,'', 'width=800,
83                     height=600');
84                 docsURLs.push((new docArray(links[count_children[i].id].desc)));
85             }
86         }
87         updateDB(activityID,inveniraStdID,'docsURLs');
88     }
89
90     //open thecnical doc on new window
91     function openDoc() {
92         window.open(linkToDT,'', 'width=800, height=600');
93     }
94
95     //get JSON file and process
96     function getData() {
97
98         console.log('https://inventivetraining.herokuapp.com/api/deployDocRequestRoute/'+i
nveniraStdID+'/'+activityID);
99
100         fetch('https://inventivetraining.herokuapp.com/api/deployDocRequestRoute/'+invenir
aStdID+'/'+activityID)
101             .then(res => res.json())
102             .then(data => processData(data))
103             .catch(function (error) {
104                 alert('Request failed', error)
105             });
106     }
107
108     //outputs last URL link from list
109     function processData(data) {
110         document.getElementById('desc').innerHTML = data.properties.summary;
111         linkToDT = data.properties.dtURL;
112         for(var i=0; i < data.properties.docsURLs.length; i++) {
113             showURLs(data.properties.docsURLs[i]);
114         }
115         createDBObject(activityID,inveniraStdID,data);
116     }
117
118     //create new object in database
119     function createDBObject(activityID,inveniraStdID,data) {
120         var obj = new Object();
121         obj.activityID = activityID;
122         obj.inveniraStdID = inveniraStdID;
123         obj.access = true;
124         obj.downloadDT = false;
125         var arr = [];
126         var docArray = function(desc, dLoad){
127             this.desc = desc;
128             this.dLoad = dLoad;
129         };
130         for (var i = 0; i < data.properties.docsURLs.length; i++) {
131             arr.push((new docArray(data.properties.docsURLs[i].desc,false)));
132         }
133         obj.docsDesc=arr;
134         myJSON = JSON.stringify(obj);
135         console.log(myJSON);
136         const options = {
137             method: 'POST',
138             headers: {
139                 'Content-type': 'application/json'
140             },
141             mode:'cors',
142             body: myJSON
143         };

```

```

142     fetch('https://inventivetraining.herokuapp.com/api/analyticsDeployDocRoute',option
143     s)
144     .then(function (response) {
145         return response.text();
146     })
147     .then(function (text) {
148         console.log('Request successful:', text);
149     })
150     .catch(function (error) {
151         log('Request failed', error)
152     });
153 }
154 //update object in database (analytics)
155 function updateDB(activityID,inveniraStdID,type) {
156     var obj = new Object();
157     obj.activityID = activityID;
158     obj.inveniraStdID = inveniraStdID;
159     obj.type=type;
160     if(type=='docsURLs')
161         obj.docsDesc = docsURLs;
162     myJSON = JSON.stringify(obj);
163     console.log(myJSON);
164     const options = {
165         method: 'PUT',
166         headers: {
167             'Content-type': 'application/json'
168         },
169         mode:'cors',
170         body: myJSON
171     };
172     fetch('https://inventivetraining.herokuapp.com/api/analyticsDeployDocRoute',option
173     s)
174     .then(function (response) {
175         return response.text();
176     })
177     .then(function (text) {
178         console.log('Request successful:', text);
179     })
180     .catch(function (error) {
181         log('Request failed', error)
182     });
183 }

```

```

1  /*
2  Projet: Projeto LEI
3  Author: Duarte Cota
4  Date: 2020
5  Script: JavaScript deploy activity Hardware configuration - test firmware
6  */
7
8  //empty fields for JSON input
9  var summary = '';
10 var codeURL = '';
11 var instrURL = '';
12 var links = [];
13 var docsLibs = [];
14 var inveniraStdID;
15 var activityID;
16
17 //aux fields
18 var count_links = 0;
19
20 //main function
21 function script() {
22     //query on URL
23     const queryString = window.location.search;
24     console.log(queryString);
25     const urlParams = new URLSearchParams(queryString);
26     inveniraStdID = urlParams.get('inveniraStdID');
27     console.log(inveniraStdID);
28     activityID = urlParams.get('activityID');
29     console.log(activityID);
30     const getCode_button = document.getElementById('getInstr_button')
31
32     getData();
33
34     getCode_button.addEventListener('click', function()
35     {
36         openDoc(instrURL);
37         updateDB(activityID, inveniraStdID, 'downloadInstr');
38     });
39
40
41     const getInstr_button = document.getElementById('getCode_button')
42     getInstr_button.addEventListener('click', function()
43     {
44         alert('Tome nota do seu ID para identificação do código: ' + inveniraStdID);
45         openDoc(codeURL);
46         updateDB(activityID, inveniraStdID, 'downloadCode');
47     });
48
49     const getLinks_button = document.getElementById('getLinks_button')
50     getLinks_button.addEventListener('click', function()
51     {
52         openLinks();
53     });
54
55 }
56
57 //shows URL description
58 function showURLs(link) {
59     links.push(link); //adiciona ao array local para abrir
60     var myDiv = document.getElementById('box3');
61     //creating checkbox element
62     var checkbox = document.createElement('input');
63     checkbox.type = 'checkbox';
64     checkbox.name = 'name';
65     checkbox.value = 'value';
66     checkbox.id = count_links;
67     //creating label for checkbox
68     var label = document.createElement('label');
69     //assigning attributes for the created label tag
70     label.htmlFor = 'id';
71     //appending the created text to the created label tag
72     label.appendChild(document.createTextNode(link.desc));
73     //appending the checkbox and label to div

```

```

74     myDiv.appendChild(checkbox);
75     myDiv.appendChild(label);
76     document.getElementById('box3').innerHTML += '<br>';
77     count_links = count_links + 1;
78 }
79
80 //open URL
81 function openLinks() {
82     var count_children = document.getElementById('box3').children;
83     docsLibs=[];
84     var docArray = function(desc){
85         this.desc = desc;
86     };
87     for(var i=0; i<count_children.length; i++) {
88         if(count_children[i].type == 'checkbox') {
89             if(count_children[i].checked) {
90                 window.open(links[count_children[i].id].lnk,', 'width=800,
91                     height=600');
92                 docsLibs.push((new docArray(links[count_children[i].id].desc)));
93             }
94         }
95     }
96     updateDB(activityID,inveniraStdID,'docsLibs');
97 }
98 //open technical doc on new window
99 function openDoc(url) {
100     window.open(url,', 'width=800, height=600');
101 }
102
103 //GET JSON file and process
104 function getData() {
105
106     console.log('https://inventivetraining.herokuapp.com/api/deployHardRequestRoute/'+
107         inveniraStdID+'/' +activityID);
108
109     fetch('https://inventivetraining.herokuapp.com/api/deployHardRequestRoute/'+inveni
110         raStdID+'/' +activityID)
111         .then(res => res.json())
112         .then(data => processData(data))
113         .catch(function (error) {
114             alert('Request failed', error)
115         });
116 }
117
118 //outputs last URL link from list
119 function processData(data) {
120     document.getElementById('desc').innerHTML = data.properties.summary;
121     summary= data.properties.summary;
122     instrURL=data.properties.instrURL;
123     codeURL=data.properties.codeURL;
124     for(var i=0; i < data.properties.libsURLs.length; i++) {
125         showURLs(data.properties.libsURLs[i]);
126     }
127     createDBObject(activityID,inveniraStdID,data);
128 }
129
130 //create new object in database
131 function createDBObject(activityID,inveniraStdID,data) {
132     var obj = new Object();
133     obj.activityID = activityID;
134     obj.inveniraStdID = inveniraStdID;
135     obj.access = true;
136     obj.downloadInstr = false;
137     obj.downloadCode = false;
138     obj.upload = false;
139     obj.studentData = [];
140     var arr = [];
141     var docArray = function(desc, dLoad){
142         this.desc = desc;
143         this.dLoad = dLoad;
144     };
145     for (var i = 0; i < data.properties.libsURLs.length; i++) {

```

```

142     arr.push((new docArray(data.properties.libsURLs[i].desc, false)));
143 }
144 obj.downloadDocsLibs=arr;
145 myJSON = JSON.stringify(obj);
146 console.log(myJSON);
147 const options = {
148     method: 'POST',
149     headers: {
150         'Content-type': 'application/json'
151     },
152     mode: 'cors',
153     body: myJSON
154 };
155
156 fetch('https://inventivetraining.herokuapp.com/api/analyticsDeployHardRoute', options)
157 .then(function (response) {
158     return response.text();
159 })
160 .then(function (text) {
161     console.log('Request successful:', text);
162 })
163 .catch(function (error) {
164     log('Request failed', error)
165 });
166
167 //update db object (analytics)
168 function updateDB(activityID, inveniraStdID, type) {
169     var obj = new Object();
170     obj.activityID = activityID;
171     obj.inveniraStdID = inveniraStdID;
172     obj.type = type;
173     if(type == 'docsLibs')
174         obj.downloadDocsLibs = docsLibs;
175     myJSON = JSON.stringify(obj);
176     console.log(myJSON);
177     const options = {
178         method: 'PUT',
179         headers: {
180             'Content-type': 'application/json'
181         },
182         mode: 'cors',
183         body: myJSON
184     };
185
186     fetch('https://inventivetraining.herokuapp.com/api/analyticsDeployHardRoute', options)
187     .then(function (response) {
188         return response.text();
189     })
190     .then(function (text) {
191         console.log('Request successful:', text);
192     })
193     .catch(function (error) {
194         log('Request failed', error)
195     });
196 }

```



```

1  /*
2  Projet: Projeto LEI
3  Author: Duarte Cota
4  Date: 2020
5  Script: Javascript InvenIRA analytics simulation - tests phase
6  */
7
8  //main function
9  function script() {
10     const selectAct = document.getElementById('options1')
11     selectAct.addEventListener('change', function()
12     {
13         var value = selectAct.value;
14         setActivity(value);
15     });
16 }
17
18 //set choosen activity
19 function setActivity(value) {
20     var url;
21     document.getElementById('name').textContent = 'ATIVIDADE: ' + value;
22     if (value == 'Consulta de documentação')
23         url = 'https://inventivetraining.herokuapp.com/api/inveniraDocRoute';
24     if (value == 'Configuração de hardware')
25         url = 'https://inventivetraining.herokuapp.com/api/inveniraHardConfigRoute';
26     getIDs(url);
27 }
28
29 //get all activityID
30 function getIDs(url) {
31     const options = {
32         method: 'GET',
33     };
34     fetch(url, options)
35     .then(res => res.json())
36     .then(data => fillIDs(data))
37     .catch(function (error) {
38         alert('Atividade inexistente', error)
39     });
40 }
41
42 //fill select box with activityID
43 function fillIDs(data){
44     var op = document.getElementById('options2')
45     var len = op.options.length;
46     if(len>0){
47         for (i = len-1; i >= 0; i--) {
48             op.options[i] = null;
49         }
50     }
51     var newOption = document.createElement('option');
52     newOption.innerHTML = '';
53     op.options.add(newOption)
54     for(var i=0; i<data.length;i++){
55         var newOption = document.createElement('option');
56         newOption.value = data[i];
57         newOption.innerHTML = data[i];
58         console.log(newOption);
59         op.options.add(newOption);
60     }
61 }
62
63 //select activityID
64 function getActID(id){
65     var act = document.getElementById(id);
66     var activityID = act.value;
67     console.log(activityID);
68     getData(activityID);
69 }
70
71 //get data for specific activityID
72 function getData(activityID) {
73     var act = document.getElementById('options1');

```

```

74     var activity = act.value;
75     if (activity == 'Consulta de documentação')
76         url = 'https://inventivetraining.herokuapp.com/api/analyticsGetDocRoute';
77     if (activity == 'Configuração de hardware')
78         url = 'https://inventivetraining.herokuapp.com/api/analyticsGetHardRoute';
79     var obj = new Object();
80     obj.activityID = activityID;
81     myJSON = JSON.stringify(obj);
82     console.log(myJSON);
83     const options = {
84         method: 'POST',
85         headers: {
86             'Content-type': 'application/json'
87         },
88         mode: 'cors',
89         body: myJSON
90     };
91     fetch(url, options)
92     .then(res => res.json())
93     .then(data => processJSON(data))
94     .catch(function (error) {
95         alert('Atividade inexistente', error)
96     });
97 }
98
99 //process response with analytics and fill table
100 function processJSON(data) {
101     console.log(data);
102     var table = document.getElementById('table');
103     for(var i = table.rows.length - 1; i > 0; i--)
104         table.deleteRow(i);
105     for(var i in data){
106         var tbl = document.getElementById('table');
107         var row = tbl.insertRow();
108         var cell1 = row.insertCell();
109         var cell2 = row.insertCell();
110         var cell3 = row.insertCell();
111         var cell4 = row.insertCell();
112         cell3.style.textAlign = 'center';
113         cell1.innerHTML = data[i].inveniraStdID;
114         for(var j in data[i].quantAnalytics){
115             cell2.innerHTML += data[i].quantAnalytics[j].name + '<br>';
116         }
117         for(var j in data[i].quantAnalytics){
118             if(data[i].quantAnalytics[j].value === true)
119                 cell3.innerHTML += 'SIM' + '<br>';
120             else if (data[i].quantAnalytics[j].value === false)
121                 cell3.innerHTML += 'NÃO' + '<br>';
122             else
123                 cell3.innerHTML += data[i].quantAnalytics[j].value + '<br>';
124         }
125         var url = data[i].qualAnalyticsURL;
126         cell4.innerHTML += '<a href=' + data[i].qualAnalyticsURL + '> ' + url +
127             '</a>'
128     }

```

ANEXO 4 : CÓDIGOS CSS FRONTEND

```
1  /*
2  Projet: Projeto LEI
3  Author: Duarte Cota
4  Date: 2020
5  Script: CSS file Inven!RA simulation
6  */
7
8  body {
9      background: whitesmoke;
10     padding: 0;
11     margin: 0;
12     overflow: scroll;
13 }
14
15 .container {
16     max-width: 950px;
17     margin: 20px auto;
18     padding: 10px 20px;
19     background: #f4f7f8;
20     border-radius: 8px;
21 }
22
23 h1 {
24     color: #003366;
25     font-family: arial, sans-serif;
26     font-size: 3em;
27     font-weight: bold;
28     padding: 1%;
29     padding-left: 40px;
30     margin-top: 0px;
31     margin-bottom: 1px;
32 }
33
34 h2 {
35     color: #003366;
36     font-family: arial, sans-serif;
37     font-size: 1.5em;
38     padding-left: 40px;
39     margin-top: 0px;
40 }
41
42 table, td, th {
43     font-family: arial, sans-serif;
44     font-size: 15px;
45     color: #003366;
46     border: 2px solid black;
47     padding-top: 5px;
48     padding-bottom: 5px;
49     padding-left: 15px;
50     padding-right: 15px;
51     position: relative;
52     margin-left: 40px;
53 }
54
55
56 #table {
57     border: 1px solid black;
58     border-collapse: collapse;
59 }
60
61 #options1, #options2 {
62     position: relative;
63     margin-left: 40px;
64     border: 2px solid #808080;
65     border-radius: 4px;
66 }
```

```

1  /*
2  Projet: Projeto LEI
3  Author: Duarte Cota
4  Date: 2020
5  Script: CSS file inventiveTr@ining
6  */
7
8  body {
9      background: #99ccff;
10     padding: 0;
11     margin: 0;
12     overflow: scroll;
13 }
14
15 .container {
16     max-width: 950px;
17     margin: 20px auto;
18     padding: 10px 20px;
19     background: #f4f7f8;
20     border-radius: 8px;
21 }
22
23 h1 {
24     color: #003366;
25     font-family: arial, sans-serif;
26     font-size: 3em;
27     font-weight: bold;
28     padding: 1%;
29     margin-top: 0px;
30     margin-bottom: 1px;
31 }
32
33 h2 {
34     color: #003366;
35     font-family: arial, sans-serif;
36     font-size: 1.5em;
37     padding-left: 1%;
38     margin-top: 0px;
39 }
40
41 h3 {
42     color: #003366;
43     font-family: arial, sans-serif;
44     font-size: 1em;
45     padding-left: 1%;
46     margin-top: 0px;
47 }
48
49 img {
50     float: right;
51     overflow: hidden;
52     width: 175px;
53     height: 175px;
54 }
55
56
57 .box {
58     font-family: arial, sans-serif;
59     background-color: #ffffff;
60     padding: 5px;
61     font-size: 12px;
62     display: inline-block;
63     margin-left: 20px
64 }
65
66 #list, #list_header, #listDoc, #listCode, #listData {
67     font-family: arial, sans-serif;
68     font-size: 12px;
69     color: #003366;
70     margin-left: 20px;
71     margin-bottom: 5px;
72 }
73

```

```

74 #title_links {
75     font-family: arial, sans-serif;
76     font-size:12px;
77     color: #003366;
78     margin-left: 20px;
79     margin-bottom: 5px;
80 }
81
82 #title, #links, #links_desc,
83 #desc_tec, #desc_code, #desc_instr {
84     font-family: arial, sans-serif;
85     font-size:12px;
86     border: 2px solid #808080;
87     border-radius: 4px;
88     outline: none;
89     margin-bottom: 10px;
90     display:inline-block;
91     position:relative;
92     margin-left: 20px;
93     resize: both;
94 }
95
96 #desc, #obj {
97     font-family: arial, sans-serif;
98     font-size:12px;
99     margin-bottom: 10px;
100     display:block;
101     position:relative;
102     margin-left: 20px;
103     border: 2px solid #808080;
104     border-radius: 4px;
105     width: 900px;
106     height: 90px;
107     min-width: 120px;
108     min-height: 90px;
109     max-width: 900px;
110     max-height: 300px;
111     resize: both;
112 }
113
114 #clear_button, #getURL_button,
115 #getLinks_button, #add_button,
116 #getInstr_button, #getCode_button,
117 #all, #last, #fileDownloadJSON, #fileDownloadCSV {
118     margin-bottom: 10px;
119     display:inline-block;
120     position:relative;
121     margin-left: 20px;
122     border: 2px solid #808080;
123     border-radius: 4px;
124 }
125
126 #rem_button {
127     margin-bottom: 10px;
128     display:inline-block;
129     position:relative;
130     margin-left: 20px;
131     border: 2px solid #808080;
132     border-radius: 4px;
133 }
134
135 .save__input {
136     display: none;
137 }
138
139 .save__button {
140     background: #808080;
141     border: 2px solid #808080;
142     border-radius: 4px;
143     outline: none;
144     padding: 0.5em 0.8em;
145     margin-left: 400px;
146     color: #ffffff;

```

```
147     font-size: 1em;
148     font-family: "Quicksand", sans-serif;
149     font-weight: bold;
150     cursor: pointer;
151     margin-bottom: 10px;
152     position: relative;
153 }
154
155 .save__button:active {
156     background: #00745d;
157 }
158
159 .save__label {
160     max-width: 250px;
161     font-size: 0.95em;
162     text-overflow: ellipsis;
163     overflow: hidden;
164     white-space: nowrap;
165     font-family: "Quicksand", sans-serif;
166 }
167
168 table, td, th {
169     font-family: arial, sans-serif;
170     font-size: 12px;
171     color: #003366;
172     border: 2px solid black;
173     padding-top: 5px;
174     padding-bottom: 5px;
175     padding-left: 15px;
176     padding-right: 15px;
177     position: relative;
178     margin: 0 auto;
179     border-collapse: collapse;
180 }
```

ANEXO 5 : CÓDIGOS BACKEND API NODE JS


```

1  /*
2  Projet: Projeto LEI
3  Author: Duarte Cota
4  Date: 2020
5  Script: Node JS API main file
6  */
7
8  const express = require("express");
9  const app = express();
10 const path = require("path");
11 const mongoose = require("mongoose");
12 const ip = require("ip");
13 var cors = require('cors')
14 console.dir (ip.address());
15
16 mongoose.set('useFindAndModify', false);
17
18 app.use(express.urlencoded({extended: true}));
19 app.use(express.json());
20 app.use(cors());
21 app.use(function(req, res, next) {
22     res.header("Access-Control-Allow-Origin", "*");
23     res.header("Access-Control-Allow-Headers", "Origin, X-Requested-With, Content-Type, Accept");
24     next();
25 });
26
27 //API routes
28 const routCourses = require("./routes/trainee_data");
29 const routeAnalyticsDepDocs = require("./routes/analyticsDeployDocRoute");
30 const routeAnalyticsDepHard = require("./routes/analyticsDeployHardRoute");
31 const routeDeployDocRequest = require("./routes/deployDocRequestRoute");
32 const routeDeployHardRequest = require("./routes/deployHardRequestRoute");
33 const routeAnalyticsGetHard = require("./routes/analyticsGetHardRoute");
34 const routeAnalyticsGetDoc = require("./routes/analyticsGetDocRoute");
35 const routeInveniraDoc = require("./routes/inveniraDocRoute");
36 const routeInveniraHardConfig = require("./routes/inveniraHardConfigRoute");
37 app.use("/api/trainee_data", routCourses);
38 app.use("/api/analyticsDeployDocRoute", routeAnalyticsDepDocs);
39 app.use("/api/analyticsDeployHardRoute", routeAnalyticsDepHard);
40 app.use("/api/deployDocRequestRoute", routeDeployDocRequest);
41 app.use("/api/deployHardRequestRoute", routeDeployHardRequest);
42 app.use("/api/analyticsGetHardRoute", routeAnalyticsGetHard);
43 app.use("/api/analyticsGetDocRoute", routeAnalyticsGetDoc);
44 app.use("/api/inveniraDocRoute", routeInveniraDoc);
45 app.use("/api/inveniraHardConfigRoute", routeInveniraHardConfig);
46
47 app.use(express.static('./public'));
48
49 //respond to deploy Documentation activity request
50 app.get('/deploydoc/:?', (req, res) => {
51     res.sendFile(path.join(__dirname, '/public/html/activity_deploy_doc.html'));
52 });
53
54 //respond to deploy Hardware Configuration activity request
55 app.get('/deployhardconfig/:?', (req, res) => {
56     res.sendFile(path.join(__dirname, '/public/html/activity_deploy_hard.html'));
57 });
58
59 //respond to Inven!RA analytics request - Documentation
60 app.get('/qualAnalyticsDoc/:?', (req, res) => {
61     res.sendFile(path.join(__dirname, '/public/html/analyticsDoc.html'));
62 });
63
64 //respond to Inven!RA analytics request - Hardware Configuration
65 app.get('/qualAnalyticsHardConfig/:?', (req, res) => {
66     res.sendFile(path.join(__dirname, '/public/html/analyticsHardConfig.html'));
67 });
68
69 //respond to simulate Inven!RA show analytics request
70 app.get('/', (req, res) => {
71     res.sendFile(path.join(__dirname, '/invenira.html'));
72 });

```

```

73
74 //respond to config Documentation activity request
75 app.get('/configDocActivity', (req, res) => {
76     res.sendFile(path.join(__dirname, '/public/html/activity_config_doc.html'));
77 });
78
79 //respond to config Hardware Configuration activity request
80 app.get('/configHardActivity', (req, res) => {
81     res.sendFile(path.join(__dirname, '/public/html/activity_config_hard.html'));
82 });
83
84 //URL to Cloud MongoDB
85 const dbURL =
86 "mongodb://dcota:MiniLeo1969@itrainingdb-shard-00-00-yh1ad.mongodb.net:27017,
87 itrainingdb-shard-00-01-yh1ad.mongodb.net:27017,itrainingdb-shard-00-02-yh1ad.mongodb.
88 net:27017/json_data?ssl=true&replicaSet=iTrainingDB-shard-0&authSource=admin&retryWrit
89 es=true&w=majority";
90
91 //connect to MongoDB
92 mongoose.connect(dbURL, {useNewUrlParser: true, useUnifiedTopology: true}, () =>
93 console.log("connected to database!"));
94
95 //define PORT
96 const PORT = 5000;
97
98 //set listen to PORT
99 app.listen(process.env.PORT || PORT, () => console.log ('Listening on port ' + PORT));

```

```
1  /*
2  Projet: Projeto LEI
3  Author: Duarte Cota
4  Date: 2020
5  Script: MongoDB schema for analytics; activity Consult documentation
6  */
7
8  const mongoose = require('mongoose');
9
10 const AnalyticsSchemaActivityDoc = mongoose.Schema({
11   'activityID': {'type': Number},
12   'inveniraStdID': {'type': Number},
13   'access': {'type': Boolean},
14   'downloadDT': {'type': Boolean},
15   'docsDesc': {'type': ['Mixed']},
16   date: {type: Date, default: Date.now}
17 });
18
19 module.exports = mongoose.model('AnalyticsDeployDoc',AnalyticsSchemaActivityDoc);
```

```
1  /*
2  Projet: Projeto LEI
3  Author: Duarte Cota
4  Date: 2020
5  Script: MongoDB schema for analytics; activity Hardware configuration
6  */
7
8  const mongoose = require('mongoose');
9
10 const AnalyticsSchemaActivityHard = mongoose.Schema({
11   'activityID': {'type': Number},
12   'inveniraStdID': {'type': Number},
13   'access': {'type': Boolean},
14   'downloadInstr': {'type': Boolean},
15   'downloadCode': {'type': Boolean},
16   'downloadDocsLibs': {'type': ['Mixed']},
17   'upload': {'type': Boolean},
18   'studentData': {'type': ['Mixed']},
19   date: {type: Date, default: Date.now}
20 });
21
22 module.exports = mongoose.model('AnalyticsDeployHard', AnalyticsSchemaActivityHard);
```

```
1  /*
2  Projet: Projeto LEI
3  Author: Duarte Cota
4  Date: 2020
5  Script: MongoDB schema for deploy; activity Consult documentation
6  */
7
8  const mongoose = require('mongoose');
9
10 const DeployDocRequestSchema = mongoose.Schema({
11   'activityID': {'type': 'Number'},
12   'inveniraStdID': {'type': 'Number'},
13   'properties': {
14     'summary': {'type': 'String'},
15     'dtURL': {'type': 'String'},
16     'docsURLs': {'type': ['Mixed']}
17   },
18   date: {type: Date, default: Date.now}
19 });
20
21 module.exports = mongoose.model('DeployDocRequest', DeployDocRequestSchema);
```

```
1  /*
2  Projet: Projeto LEI
3  Author: Duarte Cota
4  Date: 2020
5  Script: MongoDB schema for deploy; activity Hardware configuration
6  */
7
8  const mongoose = require('mongoose');
9
10 const DeployHardRequestSchema = mongoose.Schema({
11   'activityID': {'type': 'Number'},
12   'inveniraStdID': {'type': 'Number'},
13   'properties': {
14     'summary': {'type': 'String'},
15     'codeURL': {'type': 'String'},
16     'instrURL': {'type': 'String'},
17     'libsURLs': {'type': ['Mixed']}
18   },
19   date: {type: Date, default: Date.now}
20 });
21
22 module.exports = mongoose.model('DeployHardRequest', DeployHardRequestSchema);
```

```

1  /*
2  Projet: Projeto LEI
3  Author: Duarte Cota
4  Date: 2020
5  Script: Node JS API - return URL for activity deploy - Consult documentation
6  */
7
8  const express = require("express");
9  const deployDocRequest = express.Router();
10
11  const deployRequestCollection = require("../models/DeployDocRequest")
12
13  deployDocRequest.post("/", (req,res) => {
14    deployRequestCollection.find({'activityID': {$eq: req.body.activityID},
15    'inveniraStdID': {$eq: req.body.inveniraStdID}})
16    .exec()
17    .then ((data) => {
18      console.log("From database",data);
19      //if does not exist create object
20      if(data==0) {
21        const deployRequestCollectionRoute = new deployRequestCollection({
22          activityID: req.body.activityID,
23          inveniraStdID: req.body.inveniraStdID,
24          properties : req.body.properties
25        });
26        deployRequestCollectionRoute.save()
27        .then (result => {
28          console.log(result);
29        })
30        .catch(err => {
31          res.json({message: err});
32          console.log(err);
33        });
34      }
35      else{
36        var obj =
37        {"deployURL":"https://inventivetraining.herokuapp.com/deploydoc/?inveniraS
38        tdID=" + req.body.inveniraStdID + "&activityID=" + req.body.activityID }
39        res.json(obj);
40        console.log('Atividade existente.');

```

```

1  /*
2  Projet: Projeto LEI
3  Author: Duarte Cota
4  Date: 2020
5  Script: Node JS API - return URL for activity deploy - Hardware configuration,
firmware test
6  */
7
8  const express = require("express");
9  const deployHardRequest = express.Router();
10
11  //MongoDB model
12  const deployRequestCollection = require("../models/DeployHardRequest")
13
14  deployHardRequest.post("/", (req,res) => {
15      deployRequestCollection.find({'activityID': {$eq: req.body.activityID},
16      'inveniraStdID': {$eq: req.body.inveniraStdID}})
17      .exec()
18      .then ((data) => {
19          console.log("From database",data);
20          //if not exist create object
21          if(data==0) {
22              const deployRequestCollectionRoute = new deployRequestCollection({
23                  activityID: req.body.activityID,
24                  inveniraStdID: req.body.inveniraStdID,
25                  properties : req.body.properties
26              });
27              deployRequestCollectionRoute.save()
28              .then (result => {
29                  console.log(result);
30              })
31              .catch(err => {
32                  res.json({message: err});
33                  console.log(err);
34              });
35          }
36          else{
37              var obj =
38              {"deployURL":"https://inventivetraining.herokuapp.com/deployhardconfig/?in
39              veniraStdID=" + req.body.inveniraStdID + "&activityID=" +
40              req.body.activityID }
41              res.json(obj);
42              console.log('Atividade existente.');

```



```

1  /*
2  Projet: Projeto LEI
3  Author: Duarte Cota
4  Date: 2020
5  Script: Node JS API - route deploy activity Get Documentation
6  */
7
8  const express = require('express');
9  const routerAnalytics = express.Router();
10
11  //MongoDB model
12  const AnalyticsDeployDoc = require('../models/AnalyticsDeployDoc')
13
14  // POST request
15  routerAnalytics.post('/', (req,res) => {
16    AnalyticsDeployDoc.find({'activityID': {$eq: req.body.activityID},
17      'inveniraStdID': {$eq: req.body.inveniraStdID}})
18    .exec()
19    .then ((data) => {
20      console.log('From database',data);
21      //if does not exist create object
22      if(data == 0) {
23        const analyticsDeployDocRoute = new AnalyticsDeployDoc({
24          activityID: req.body.activityID,
25          inveniraStdID: req.body.inveniraStdID,
26          access : req.body.access,
27          downloadDT : req.body.downloadDT,
28          docsDesc : req.body.docsDesc
29        });
30
31        analyticsDeployDocRoute.save()
32        .then (result => {
33          res.json(result);
34          console.log(result);
35        })
36        .catch(err => {
37          res.json({message: err});
38          console.log(err);
39        });
40      }
41      else{
42        console.log('StudentID existente.');
```

```

43        console.log('Objeto criado com sucesso...');
44      }
45    })
46    .catch ((error) => {
47      console.log(error);
48      res.status(500).json({error: error});
49    });
50  });
51
52  //PUT request
53  routerAnalytics.put('/', (req,res) => {
54    if(req.body.type=='dtURL'){
55      AnalyticsDeployDoc.findOneAndUpdate({'activityID': {$eq:
56        req.body.activityID}, 'inveniraStdID': {$eq: req.body.inveniraStdID}}, {$set
57        : {'downloadDT': true}}, {new:true})
58      .exec()
59      .then ((data) => {
60        if(data != []) {
61          res.status(200).json(data);
62          console.log('From database',data);
63        }
64        else{
65          res.json({message:'Objeto inexistente.'});
66          console.log('Objeto inexistente.');
```

```

67        }
68      })
69      .catch ((error) => {
70        console.log(error);
71        res.status(500).json({error: error});
72      });
73    }
74  });
75

```

```

71     else {
72         var len = req.body.docsDesc.length;
73         for(var i=0 ; i < len ;i++){
74             var doc = req.body.docsDesc[i];
75             console.log(doc.desc);
76             if(doc!=null){
77                 AnalyticsDeployDoc.findOneAndUpdate({'activityID': {$eq:
78                     req.body.activityID}, 'inveniraStdID': {$eq:
79                     req.body.inveniraStdID}, 'docsDesc.desc':{$eq: doc.desc} }, {$set :
80                     {'docsDesc.$.dLoad': true}}, {new:true})
81                 .exec()
82                 .then ((data) => {
83                     if(data != []) {
84                         console.log('From database',data);
85                     }
86                     else{
87                         console.log('Objeto inexistente.');

```

```

1  /*
2  Projet: Projeto LEI
3  Author: Duarte Cota
4  Date: 2020
5  Script: Node JS API - route deploy activity Hardware configuration - test firmware
6  */
7
8  const express = require('express');
9  const routerAnalytics = express.Router();
10
11  //MongoDB model
12  const AnalyticsDeployHard = require('../models/AnalyticsDeployHard')
13
14  //POST request
15  routerAnalytics.post('/', (req,res) => {
16    AnalyticsDeployHard.find({'activityID': {$eq: req.body.activityID},
17      'inveniraStdID': {$eq: req.body.inveniraStdID}})
18    .exec()
19    .then ((data) => {
20      console.log('From database',data);
21      //if does not exist create object
22      if(data == 0) {
23        const analyticsDeployHardRoute = new AnalyticsDeployHard({
24          activityID: req.body.activityID,
25          inveniraStdID: req.body.inveniraStdID,
26          access : req.body.access,
27          downloadInstr : req.body.downloadInstr,
28          downloadCode : req.body.downloadCode,
29          downloadDocsLibs : req.body.downloadDocsLibs,
30          upload : req.body.upload,
31          studentData: req.body.studentData
32        });
33
34        analyticsDeployHardRoute.save()
35        .then (result => {
36          console.log(result);
37          console.log('Objeto criado com sucesso...');
38        })
39        .catch(err => {
40          res.json({message: err});
41          console.log(err);
42        });
43      }
44      else{
45        console.log('StudentID existente.');

```

```

71     });
72 }
73 else if(req.body.type=='downloadCode'){
74     AnalyticsDeployHard.findOneAndUpdate({'activityID': {$eq:
75     req.body.activityID}, 'inveniraStdID': {$eq: req.body.inveniraStdID}},
76     {$set: {'downloadCode': true}}, {new:true})
77     .exec()
78     .then ((data) => {
79         if(data != []) {
80             res.status(200).json(data);
81             console.log('From database',data);
82         }
83         else{
84             res.json({message:'Objeto inexistente.'});
85             console.log('Objeto inexistente.');

```

```

1  /*
2  Projet: Projeto LEI
3  Author: Duarte Cota
4  Date: 2020
5  Script: Node JS API - return analytics activity Get Documentation
6  */
7
8  const express = require('express');
9  const routerAnalytics = express.Router();
10
11  //MongoDB model
12  const AnalyticsGetDoc = require('../models/AnalyticsDeployDoc')
13
14  //POST request
15  routerAnalytics.post('/', (req,res) => {
16    AnalyticsGetDoc.find({'activityID': {$eq: req.body.activityID}})
17    .exec()
18    .then ((data) => {
19      console.log('From database',data);
20      if(data != 0) {
21        var maxInteractions = data[0].docsDesc.length + 2;
22        var getAnalytics = [];
23        for(var i in data){
24          var objects = [];
25          var count=0;
26          var totalInteractions=0;
27          if(data[i].access==true)
28            totalInteractions+=1;
29          if(data[i].downloadDT==true)
30            totalInteractions+=1
31          for(var j in data[i].docsDesc){
32            if(data[i].docsDesc[j].dLoad == true)
33              count = count + 1;
34          }
35          var percDocsTec = (count/data[i].docsDesc.length)*100;
36          var percProgress=((totalInteractions+count)/maxInteractions)*100
37          var obj = {'name': 'Acesso à atividade', 'type': 'boolean', 'value':
38            data[i].access}
39            objects.push(obj);
40            obj = {'name': 'Download Descritivo Técnico', 'type': 'boolean',
41              'value': data[i].downloadDT};
42            objects.push(obj);
43            obj = {'name': 'Download Documentos Técnicos (%)', 'type':
44              'percentage', 'value': percDocsTec.toFixed(2)};
45            objects.push(obj);
46            obj = {'name': 'Progresso na atividade (%)', 'type': 'percentage',
47              'value': percProgress.toFixed(2)};
48            objects.push(obj);
49            obj = {'inveniraStdID': data[i].inveniraStdID, 'quantAnalytics':
50              objects,
51
52              'qualAnalyticsURL':'https://inventivetraining.herokuapp.com/qualAn
53              alyticsDoc/?activityID='+ req.body.activityID + '&'
54              +'inveniraStdID=' +data[i].inveniraStdID }
55            getAnalytics.push(obj);
56          }
57          res.json(getAnalytics);
58        }
59      }
60      else{
61        console.log('Atividade sem analíticas para mostrar');
62        var obj = {'status': 'Atividade sem analíticas para mostrar'};
63        res.json(obj);
64      }
65    })
66    .catch ((error) => {
67      console.log(error);
68      res.status(500).json({error: error});
69    });
70  });
71
72  //GET request
73  routerAnalytics.get('/:activityID/:inveniraStdID', (req,res) => {

```

```

66 AnalyticsGetDoc.findOne({'activityID': {$eq: req.params.activityID},
67 'inveniraStdID': {$eq: req.params.inveniraStdID}})
68 .exec()
69 .then ((data) => {
70     console.log('From database',data);
71     if(data != 0) {
72         var arr=[];
73         if(data.downloadDT==true)
74             arr.push('Descritivo Técnico')
75         var len = data.docsDesc.length;
76         for(var i=0; i<len; i++){
77             if(data.docsDesc[i].dLoad==true)
78                 arr.push(data.docsDesc[i].desc)
79         }
80         res.json(arr);
81     }
82     else{
83         console.log('Base de dados vazia');
84         res.send('Base de dados vazia');
85     }
86 })
87 .catch ((error) => {
88     console.log(error);
89     res.status(500).json({error: error});
90 });
91
92 module.exports = routerAnalytics;

```

```

1  /*
2  Projet: Projeto LEI
3  Author: Duarte Cota
4  Date: 2020
5  Script: Node JS API - return analytics activity Hardware configuration - test firmware
6  */
7
8  const express = require('express');
9  const routerAnalytics = express.Router();
10
11  //MongoDB model
12  const AnalyticsGetHard = require('../models/AnalyticsDeployHard')
13
14  //POST request
15  routerAnalytics.post('/', (req,res) => {
16    AnalyticsGetHard.find({'activityID': {$eq: req.body.activityID}})
17    .exec()
18    .then ((data) => {
19      console.log('From database',data);
20      var getAnalytics = [];
21      if(data != 0) {
22        var maxInteractions = data[0].downloadDocsLibs.length + 4;
23        for(var i in data){
24          var objects = [];
25          var count=0;
26          var totalInteractions=0;
27          if(data[i].access==true){
28            totalInteractions +=1;
29          }
30          if(data[i].downloadInstr==true){
31            totalInteractions += 1;
32          }
33          if(data[i].downloadCode==true) {
34            totalInteractions += 1;
35          }
36          if(data[i].upload==true){
37            totalInteractions += 1;
38          }
39          for(var j in data[i].downloadDocsLibs){
40            if(data[i].downloadDocsLibs[j].dLoad == true)
41              count += 1;
42          }
43          var percDocsTec = (count/data[i].downloadDocsLibs.length)*100;
44          var percProgress=((totalInteractions+count)/maxInteractions)*100;
45          var obj = {'name': 'Acesso à atividade', 'type': 'boolean', 'value':
46            data[i].access}
47            objects.push(obj);
48            obj = {'name': 'Download Instruções', 'type': 'boolean', 'value':
49              data[i].downloadInstr};
50            objects.push(obj);
51            obj = {'name': 'Download Código-Base', 'type': 'boolean', 'value':
52              data[i].downloadCode};
53            objects.push(obj);
54            obj = {'name': 'Upload dados', 'type': 'boolean', 'value':
55              data[i].upload};
56            objects.push(obj);
57            obj = {'name': 'Download Documentos Técnicos (%)', 'type':
58              'percentage', 'value': percDocsTec.toFixed(2)};
59            objects.push(obj);
60            obj = {'name': 'Progresso na atividade (%)', 'type': 'percentage',
61              'value': percProgress.toFixed(2)};
62            objects.push(obj);
63            obj = {'inveniraStdID': data[i].inveniraStdID, 'quantAnalytics':
64              objects,
65              'qualAnalyticsURL': 'https://inventivetraining.herokuapp.com/qualAn
66                alyticsHardConfig/?activityID='+ req.body.activityID + '&
67                +inveniraStdID=' +data[i].inveniraStdID }
68            getAnalytics.push(obj);
69          }
70          res.json(getAnalytics);
71        }
72      }
73    }
74  }
75  }
76  }
77  }
78  }
79  }
80  }
81  }
82  }
83  }
84  }
85  }
86  }
87  }
88  }
89  }
90  }
91  }
92  }
93  }
94  }
95  }
96  }
97  }
98  }
99  }
100 }

```

```

64         console.log('Atividade sem analíticas para mostrar.');
```

```

65         var obj = {'status': 'Atividade sem analíticas para mostrar'};
```

```

66         res.json(obj);
```

```

67     }
68 })
69 .catch ((error) => {
70     console.log(error);
71     res.status(500).json({error: error});
72 });
73 });
74
75 //GET request
76 routerAnalytics.get('/:activityID/:inveniraStdID', (req,res) => {
77     AnalyticsGetHard.findOne({'activityID': {$eq: req.params.activityID},
78     'inveniraStdID': {$eq: req.params.inveniraStdID}})
79     .exec()
80     .then ((data) => {
81         console.log('From database',data);
82         if(data != 0) {
83             var arrDoc=[];
84             var arrCode=[];
85             var arrUpload=[];
86             if(data.downloadInstr==true)
87                 arrDoc.push('Instruções do projeto');
88             if(data.downloadCode==true)
89                 arrCode.push('Código-base');
90             var len = data.downloadDocsLibs.length;
91             for(var i=0; i<len; i++){
92                 if(data.downloadDocsLibs[i].dLoad==true)
93                     arrCode.push(data.downloadDocsLibs[i].desc)
94             }
95             if(data.studentData.length>0)
96                 arrUpload = data.studentData;
97             var obj = {'downloadInstr': arrDoc, 'downloadCode' : arrCode,
98             'studentData': arrUpload};
99             console.log(obj);
100             res.json(obj);
101         }
102         else{
103             console.log('Base de dados vazia');
104             res.send('Base de dados vazia');
105         }
106     })
107     .catch ((error) => {
108         console.log(error);
109         res.status(500).json({error: error});
110     });
111 });
112
113 module.exports = routerAnalytics;
```



```

1  /*
2  Projet: Projeto LEI
3  Author: Duarte Cota
4  Date: 2020
5  Script: Node JS API - Inven!RA analytics simulation - Consult documentation
6  */
7
8  const express = require('express');
9  const routerAnalytics = express.Router();
10
11  //MongoBD model
12  const AnalyticsGetDoc = require('../models/AnalyticsDeployDoc')
13
14  routerAnalytics.get('/', (req,res) => {
15    AnalyticsGetDoc.find()
16    .exec()
17    .then ((data) => {
18      console.log('From database',data);
19      if(data != 0) {
20        var objects=[];
21        for(var i in data){
22          if(objects.includes(data[i].activityID,0)==false)
23            objects.push(data[i].activityID);
24        }
25        console.log(objects);
26        res.json(objects);
27      }
28      else{
29        console.log('Base de dados vazia');
30        res.send('Base de dados vazia');
31      }
32    })
33    .catch ((error) => {
34      console.log(error);
35      res.status(500).json({error: error});
36    });
37  });
38
39  module.exports = routerAnalytics;

```

```

1  /*
2  Projet: Projeto LEI
3  Author: Duarte Cota
4  Date: 2020
5  Script: Node JS API - Inven!RA analytics simulation - Hardware configuration, test
   firmware
6  */
7
8  const express = require('express');
9  const routerAnalytics = express.Router();
10
11  //MongoDB model
12  const AnalyticsGetHard = require('../models/AnalyticsDeployHard')
13
14  routerAnalytics.get('/', (req,res) => {
15      AnalyticsGetHard.find()
16      .exec()
17      .then ((data) => {
18          console.log('From database',data);
19          if(data != 0) {
20              var objects=[];
21              for(var i in data){
22                  if(objects.includes(data[i].activityID,0)==false)
23                      objects.push(data[i].activityID);
24              }
25              console.log(objects);
26              res.json(objects);
27          }
28          else{
29              console.log('Base de dados vazia');
30              res.send('Base de dados vazia');
31          }
32      })
33      .catch ((error) => {
34          console.log(error);
35          res.status(500).json({error: error});
36      });
37  });
38
39  module.exports = routerAnalytics;

```

ANEXO 6 : CÓDIGO-BASE + CÓDIGO TESTE HARDWARE

```

1  /*
2
3  UFCD6073: Microcontroladores-aplicações
4  Ficheiro: Test code
5  */
6
7  /*
8   * 1- Check if you have all libs installed in your computer
9   * 2- You may include your own libs
10  * 3- Please declare your variables and objects in myDeclarations
11  * 4- Create your data in function myData()
12  * */
13
14  //replace VALUE with the supplied value when downloading this code
15  #define ID 1001
16
17  #include <Wire.h>
18  #include <SPI.h>
19  #include <Ethernet.h>
20  #include <ArduinoJson.h>
21  #include <Adafruit_BMP280.h>
22  #include <Adafruit_Sensor.h>
23  #include <DHT.h>
24
25  //define your sensor pins here
26  #define DHTPIN 3
27  #define DHTTYPE DHT22
28  //define your setpoint for temperature alarm here
29  #define SETPOINT 25
30
31  int count = 1;
32  byte mac[] = {0xDE, 0xAD, 0xBE, 0xEF, 0xEF, 0xED};
33  byte APserver[] = {192, 168, 1, 88};
34  unsigned long previousMillis = 0;
35  int interval = 1000;
36
37  //do not edit next two lines
38  const size_t capacity = 5*JSON_ARRAY_SIZE(10) + JSON_OBJECT_SIZE(6);
39  DynamicJsonDocument doc(capacity);
40
41  //define JSON structures here
42  JsonArray temperature = doc.createNestedArray("Temperatura(°C)");
43  JsonArray pressure = doc.createNestedArray("Pressão(hPa)");
44  JsonArray humidity = doc.createNestedArray("Humidade(%)");
45  JsonArray al = doc.createNestedArray("Alarme(ON/OFF)");
46  JsonArray sp = doc.createNestedArray("Setpoint(°C)");
47
48
49  //*****
50  void createJsonData(){
51      doc["inveniraStdID"] = ID;
52      while(count<=10){
53          unsigned long currentMillis = millis();
54          if (currentMillis - previousMillis >= interval){
55              previousMillis=currentMillis;
56              Serial.println(count);
57              myData(); //student data
58              count++;
59          }
60      }
61  }
62
63  //this function is for your code!
64  void myData(){
65      Adafruit_BMP280 bmp;
66      DHT dht(DHTPIN, DHTTYPE);
67      bmp.begin();
68      dht.begin();
69      float temp;
70      float pre;
71      float hum;
72      int alarm;
73      temp=(bmp.readTemperature());

```

```

74     temperature.add(temp);
75     pre=bmp.readPressure()/100;
76     pressure.add(pre);
77     hum=dht.readHumidity();
78     humidity.add(hum);
79     if(temp>SETPOINT)
80         al.add(1);
81     else
82         al.add(0);
83     sp.add(SETPOINT);
84 }
85 //*****
86
87 void setup() {
88     // Open serial communications and wait for port to open:
89     Serial.begin(9600);
90     //update your ethernet shield IP here...
91     byte ip[] = {192, 168, 1, 105};
92     EthernetClient client;
93     Ethernet.begin(mac,ip);
94     createJsonData();
95     serializeJsonPretty(doc, Serial);
96     Serial.println();
97     Serial.println("connecting...");
98     if(client.connect(APserver, 5000)){
99         Serial.println("Connected to server");
100         Serial.println("Begin POST Request");
101         client.println("PUT /api/analyticsDeployHardRoute HTTP/1.1");
102         client.println("Host: localhost:5000");
103         client.println("Access-Control-Allow-Origin: *");
104         client.println("Content-Type: application/json");
105         client.println("Connection: close");
106         client.print("Content-Length: ");
107         client.println(measureJson(doc));
108         client.println();
109         serializeJson(doc, client);
110         Serial.println("Request completed");
111     }
112     else
113         Serial.println("Connection failed, check your local IP address and try
again...");
114 }
115
116 void loop() {
117     //do not use this function
118 }

```

```

1  /*
2  UFCD6073: Microcontroladores-aplicações
3  Ficheiro: Base code
4  */
5
6  /*
7   * 1- Check if you have all libs installed in your computer
8   * 2- You may include your own libs
9   * 3- Please declare your variables and objects in myDeclarations
10  * 4- Create your data in function myData()
11  * */
12
13  //replace VALUE with the supplied value when downloading this code
14  #define ID VALUE
15
16  #include <Wire.h>
17  #include <SPI.h>
18  #include <Ethernet.h>
19  #include <ArduinoJson.h>
20  #include <Adafruit_BMP280.h>
21  #include <Adafruit_Sensor.h>
22  #include <DHT.h>
23
24  //define your sensor pins here
25  #define DHTPIN 3
26  #define DHTTYPE DHT22
27  //define your setpoint for temperature alarm here
28  #define SETPOINT 25
29
30  /*not edit this lines*/
31  int count = 1;
32  byte mac[] = {0xDE, 0xAD, 0xBE, 0xEF, 0xEF, 0xED};
33  byte APserver[] = {192, 168, 1, 88};
34  unsigned long previousMillis = 0;
35  int interval = 1000;
36  const size_t capacity = 5*JSON_ARRAY_SIZE(10) + JSON_OBJECT_SIZE(6);
37  DynamicJsonDocument doc(capacity);
38  /*****/
39
40  //define JSON structures here
41  JsonArray temperature = doc.createNestedArray("Temperatura(°C)");
42  JsonArray pressure = doc.createNestedArray("Pressão(hPa)");
43  JsonArray humidity = doc.createNestedArray("Humidade(%)");
44  JsonArray al = doc.createNestedArray("Alarme(ON/OFF)");
45  JsonArray sp = doc.createNestedArray("Setpoint(°C)");
46
47  void createJsonData(){
48      doc["inveniraStdID"] = ID;
49      while(count<=10){
50          unsigned long currentMillis = millis();
51          if (currentMillis - previousMillis >= interval){
52              previousMillis=currentMillis;
53              Serial.println(count);
54              myData(); //student data
55              count++;
56          }
57      }
58  }
59
60  //this function is for your code!
61  void myData(){
62      //your code
63  }
64  /*****
65
66  void setup() {
67      // Open serial communications and wait for port to open:
68      Serial.begin(9600);
69      //update your ethernet shield IP here...
70      byte ip[] = {192, 168, 1, 105};
71      EthernetClient client;
72      Ethernet.begin(mac,ip);
73      createJsonData();

```

```

74     serializeJsonPretty(doc, Serial);
75     Serial.println();
76     Serial.println("connecting...");
77     if(client.connect(APserver, 5000)){
78         Serial.println("Connected to server");
79         Serial.println("Begin POST Request");
80         client.println("PUT /api/analyticsDeployHardRoute HTTP/1.1");
81         client.println("Host: localhost:5000");
82         client.println("Access-Control-Allow-Origin: *");
83         client.println("Content-Type: application/json");
84         client.println("Connection: close");
85         client.print("Content-Length: ");
86         client.println(measureJson(doc));
87         client.println();
88         serializeJson(doc, client);
89         Serial.println("Request completed");
90     }
91     else
92         Serial.println("Connection failed, check your local IP address and try
again...");
93 }
94
95 void loop() {
96     //do not use this function
97 }

```